

AD-A063 167

ARMY ENGINEER WATERWAYS EXPERIMENT STATION VICKSBURG MISS. F/G 9/2  
GRAPHICS COMPATIBILITY SYSTEM (GCS). PRIMER ON COMPUTER GRAPHIC--ETC(U)  
1978

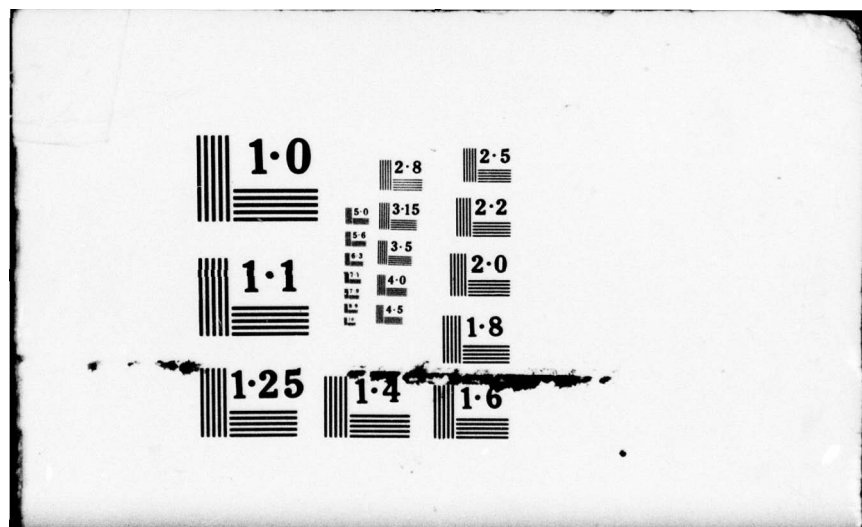
UNCLASSIFIED

NL

1 OF 5  
AD A  
063167

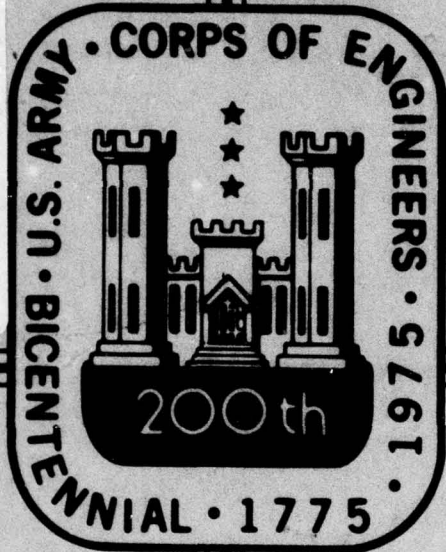






DDC FILE COPY

ADA063167



LEVEL II

WATERWAYS <sup>2</sup>  
EXPERIMENT  
STATION

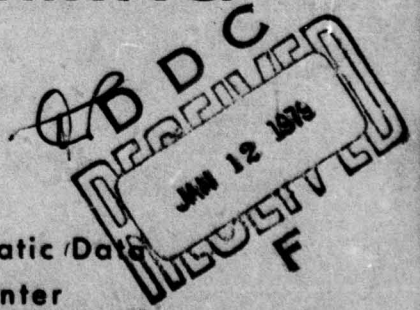
Vicksburg, Miss.

This document has been approved  
for public release and sale; its  
distribution is unlimited.

This program is furnished by the Government and is accepted and used by the recipient with the express understanding that the United States Government makes no warranties, expressed or implied, concerning the accuracy, completeness, reliability, useability, or suitability for any particular purpose of the information and data contained in this program or furnished in connection therewith, and the United States shall be under no liability whatsoever to any person by reason of any use made thereof. The program belongs to the Government. Therefore, the recipient further agrees not to assert any proprietary rights therein or to represent this program to anyone as other than a Government program.

PRIMER ON  
COMPUTER  
GRAPHICS  
PROGRAMMING

for the WES Automatic Data  
Processing Center



~~FOUO~~  
1273 01 11 233

## ELECTRONIC COMPUTER PROGRAM ABSTRACT

TITLE OF PROGRAM		PROGRAM NO.	
Graphics Compatibility System (GCS)		803-F3-R0200	
PREPARING AGENCY U. S. Army Engineer Waterways Experiment Station, Automatic Data Processing Center, P. O. Box 631, Vicksburg, Miss. 39180			
AUTHOR(S)		DATE PROGRAM COMPLETED	STATUS OF PROGRAM
West Point Military Academy		1975 <sup>(11)</sup>	PHASE
Waterways Experiment Station (ADPC)		Revised 1978	STAGE OP
A. PURPOSE OF PROGRAM			
<p>GCS is a general-purpose, device-independent computer graphics package keyed to the user.</p> <p>⑥ Graphics Compatibility System (GCS). Primer on Computer Graphics Programming. Programmers Reference Manual.</p>			
B. PROGRAM SPECIFICATIONS			
<p>GCS is a collection of ANSI standard FORTRAN subroutines invocable by a user's FORTRAN program.</p>			
C. METHODS			
<p>GCS's capabilities range over a powerful set of functions. It can produce a simple line-drawing (with over 40 types of lines), or handle comprehensive general-purpose axis-creation and automatic clipping; ARC and Conic generation; graphics input; drafting multilevel, secondary-coordinate system definition and characters; and italicized software characters. All plotting and other positional output can be done in the user's own units without his having to know anything about the specific nature of his terminal.</p>			
D. EQUIPMENT DETAILS			
<p>GCS is operational on the Honeywell 635 and supports the Tektronix 4010/4014, Computer 400/15, IMLAC PDS-1D, TSP Analog pen plotter, Datapoint 3300, and alphanumeric printing terminals. Nationwide field testing has included, or will include, such equipment as the Univac 1108, IBM S/360-370, DEC PDP 10, HP 3000, and CDC 3500 and 6000 series. A tape must be provided by interested parties if a copy of the GCS system is desired.</p>			
E. INPUT-OUTPUT			
<p>Input-Output is accomplished by invoking GCS subroutines.</p> <p><del>Supervisor</del> hb AD 59 923</p>			
F. ADDITIONAL REMARKS			
<p>Manuals available:</p> <ol style="list-style-type: none"> <li>1. <u>Primer on Computer Graphics Programming.</u></li> <li>2. <u>GCS Programmer's Reference Manual.</u></li> </ol>			

038100 79 01 11 23/13



## TABLE OF CONTENTS

Chapter I	Introduction
Chapter II	Important Technical Concepts and Conventions
Chapter III	GCS Programming Fundamentals
Chapter IV	Virtual and Device Graphics
Chapter V	Alphanumeric Output
Chapter VI	Graphical and Alphanumeric Input
Chapter VII	GCS Utility Subroutines
Chapter VIII	High Level Graphics
Chapter IX	Coordinate Systems and Transformations
Chapter X	Three Dimensional Graphics
Chapter XI	Graphical Data Structure Processing
	Picture Segmentation and Naming
Appendix A	Alphabetic Listing of GCS Subroutines
Appendix B	USET Options
Appendix C	UPSET Options
Appendix D	GCS Default Conditions
Appendix E	USET Options by Class
Appendix F	UQUERY Options
Appendix G	GCS Error Codes

ACCESSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
<i>Letter on file</i>	
BY	
DISTRIBUTION/AVAILABILITY CODES	
D:	SPECIAL
<i>R</i>	

## INTRODUCTION

### THE GCS RESEARCH AND DEVELOPMENT PROJECT

The traditional forms of computer output — vast piles of densely printed computer listings — often leave managers, engineers and students confused and frustrated. Often the information one wants to know is somewhere in the computer output, but not in a form wherein the user can find what he wants, understands, or can easily assimilate.

For years the computer field has looked forward to breaking this bottleneck in computer-human communications by using pictures instead of printouts. The idea is not new. The first stored-program, electronic computer ever designed had in its initial design an oscilloscope which was supposed to give graphic displays of the solutions of the problems it was to solve. Working two-way man-to-computer-to-man graphic communications dates back at least 10-20 years to the early Cape Cod and SAGE Air Defense Systems and Ivan Sutherland's early SKETCHPAD system. Yet, even today, the amount of graphic output that most computer users ever see is limited to a few crude barcharts of graphs produced on computer printers, or possibly a pen-plot made with an electromechanical plotter. Everyone knows about computer graphics — but very few actually have it, and even fewer make good use of it in their normal day-to-day work.

Computer graphics has been too expensive. The hardware has been expensive but, far worse than that, the software has been atrociously expensive. It has been expensive not only in initial cost, but also in its tendency to be both highly specialized and highly machine-dependent. Only specially qualified assembly language programmers could work on it and it would work on only one machine. Obviously there were exceptions to this. Much of the great acceptance with pen-plotters received in recent years came from the fact that they permitted the use of FORTRAN subroutines which could be used by run-of-the-mill FORTRAN programmers to plot simple axes, graphs, etc. The more recent success of the TEKTRONIX storage tube terminals has also been greatly influenced by the fact that reasonably good user-accessible software is available for such devices.

Today we are in a period of rapid improvement in availability of practical computer graphics. Timesharing is bringing the computer out of the warrens of the computer center and into the offices of engineers, managers, classrooms, academic buildings, and even the dormitories of academia. Timesharing started out with printing terminals, like the teletype, but the era of timesharing graphics is fast approaching. LSI and MOS technology are making the minicomputer and even the microcomputer increasingly attractive. Minicomputer-based special purpose graphics systems are now very successfully paying their own way in a number of tasks such as in the layout of electronic circuit boards. Coupled with small mass storage devices such as discs, or even in some cases cassettes, minicomputer-based graphics systems have considerable capability, in spite of serious software limitations. Today they seem ready to burst out into widespread use. Perhaps the most exciting possibility of all is the intelligent terminal connected by a communications link to a timesharing or other large computer system. In such an environment, the software advantages of the large system and the hardware advantages of the small system can complement one another, and can — potentially at least — gain the advantages of both.

But, enough of the past and the future. What can GCS offer today to the typical user — e.g., a routine FORTRAN programmer at an installation which has a limited amount of money to invest in graphics?

→ The USMA Graphics Compatibility System (GCS) is a FORTRAN-based computer graphics system designed for interactive use on a wide variety of computer graphics terminals. Due to its comprehensive and modular design, GCS provides a simplified easy-to-learn and easy-to-use approach to computer graphics, while simultaneously



providing a powerful tool which the sophisticated programmer may use for demanding and highly interactive graphical applications. ↖

GCS provides compatibility at two distinct levels:

- A. Cross-compatibility from one computer systems to another; and,
- B. Cross-compatibility between computer graphics terminals.

Cross-compatibility from one computer to another has been achieved by writing all GCS software in FORTRAN, with emphasis placed on strict adherence to the specifications established by the American National Standards Institute (ANSI).

Cross-compatibility from one terminal to another has been achieved in a manner which is completely transparent to the user. One need no longer be concerned about the problems of "tailoring" his programs for a given graphical device — GCS provides the maximum capabilities offered by the various terminals in order to satisfy the user's graphical requirements. Device independence provides the user with the option of preparing and debugging his graphics program on a terminal other than the type he may desire for final output — a consideration of paramount importance for installations which may have a limited number of graphical devices available at any given time.

It should be emphasized that GCS is not just another collection of unrelated graphics subroutines — it is a unified system which provides the user with the flexibility to perform his tasks at any level of involvement that he desires. Because of this unique man/software relationship, GCS can also be thought of as "user compatible," and it is in this last facet of compatibility that GCS unveils its true potential. Individuals who would normally dismiss graphics for their particular applications are now provided with an alternative. High-level, composite routines are provided to perform relatively complex but frequently required tasks such as the preparation of complete graphs and histograms. Flexibility and adaptability to the needs of the more sophisticated user are achieved by allowing such a user to control the various GCS options which are present to standard default conditions for the unsophisticated programmer. Users of all disciplines may feed equally comfortable with GCS, since they are required to interact only at their particular level of graphics involvement.

The GCS development was a response to a specific need. In the winter of 1971-72, it was decided that the U.S. Military Academy would teach a series of courses to senior R&D managers dealing with computer assisted design-engineering, with special emphasis upon interactive computer graphics.

It was decided that the course would be heavily, hands-on oriented and that USMA would demonstrate practical applications of graphics from each Math-Science-Engineering academic department area, programmed by a faculty member from that area. It was also decided that these applications should be spread over at least four types of graphics output:

1. Printer-plot graphics — both in batch mode and from a timeshare terminal.
2. Pen-plot graphics — both interactive and non-interactive.
3. Static or add-on CRT graphics — storage tube display.
4. Dynamic CRT graphics — refresh tube display.

The Army Material Command (AMC) was to supply adjunctive graphics equipment to add to the existing USMA Academic Computer Center timesharing systems: USMA was to provide the teaching faculty and develop both the educational software and the required

computer software; the U.S. Army Computer Systems Command was to provide a Liaison Officer who would assist in the computer software development.

The course was to begin in July 1972. The new graphics equipment would not be available until April or May. Terminal equipment would arrive so late that programs to run on one type of graphics terminal might have to be checked out by testing it on another type of terminal.

The academic department faculty members would not be available to work on the project until the start of June. Most of them were reasonably competent FORTRAN programmers with little, if any, prior experience with graphics — but most of them would have time to take a 6-8 lesson course in graphics during their lunch periods once a week in the spring semester.

In order to meet this near-impossible set of constraints, a system was devised which could be both quickly and easily learned by faculty members with varied levels of FORTRAN experience (and, in many cases, no FORTRAN experience) and could be built and checked out in a tremendous hurry, using no more than one or two full-time programmers, with part-time assistance of half a dozen or so other programmers.

GCS (Version 1.0) was the response. Somehow it was developed in time. It sputtered, wheezed, ran slowly, did not have very many sophisticated options, and had embedded in it various pieces of proprietary software which were not releasable to others. It was well accepted, however, by its users. It met the seminar requirements and was widely used during the Academic Year 1972-73, with a great deal of useful feedback being obtained. A year later, in the late spring of 1973, GCS version 2.0, rebuilt from the bottom up, became operational. It still ran the same program, with occasional minor changes suggested by users as human engineering improvements. It was faster, supported more advanced options, was much more solid, and no longer contained even the most remote trace of any proprietary software in its modules.

In October 1973, GCS had progressed to the stage that it was ready to release for field testing by other agencies. As a result of this field testing, some interesting and valuable suggestions were incorporated into GCS. In January 1975, the U.S. Military Academy was no longer able to support GCS and the responsibility for maintenance, distribution, and future development was transferred to the U.S. Army Corps of Engineers Waterways Experiment Station.



## CHAPTER I

### IMPORTANT TECHNICAL CONCEPTS AND CONVENTIONS

In order to understand and use a system as powerful and versatile as GCS, the user should be aware of certain basic technical concepts and conventions.

#### Relationship of GCS to FORTRAN

The Graphics Compatibility System (GCS) is a package of inter-related subroutines written in ANSI FORTRAN. In order to produce graphics output, a FORTRAN program is written to perform whatever computations and/or graphics manipulations are needed to produce the desired image. Embedded in the program are calls to GCS subroutines to handle the graphic display and interaction, if relevant.

Under normal conditions, all input-output between the computer and graphics devices such as plotters and terminals will be handled via calls to GCS routines, without recourse to system dependent routines.

The programming conventions of GCS are identical to those of ANSI FORTRAN except that the use of GCS creates additional reserved names. All GCS subroutines designed for *User* use are FORTRAN subroutines beginning with the letter *U*. The GCS system also includes additional internal subroutines not intended for availability to users which always begin with the letters *GCS*. Thus the user, to avoid potential confusion and/or conflict between his program and GCS, should avoid creating his own subroutines with names beginning with either *U* or *GCS*.

All numbers used as calling parameters are passed to GCS as real numbers, even in situations where they represent integers.

#### Graphics Status Area (GSA) - Preset Modes and Parameters

In order to keep continuous track of many items of information relevant to graphics activities such as where the beam (or pen) is currently located, what portion of the screen (or plotting area) is currently within limits and what portion is currently off limits for drawing, what portion is within limits and what portion is off limits for textual material, etc., a labelled COMMON area GSA is established. The GSA common block also keeps track of the user's current choice from among the list of available options which define or mode of operation of the system.

When one starts a GCS program, the user's start routine (*USTART*) automatically sets all options to a standard initial condition: rectangular coordinate system, absolute (rather than relative or incremental), plotting angles to be measured in degrees, lines to be drawn routinely as solid lines without special attributes, etc.

The following is a sample program which illustrates the use of the GCS package to produce graphics output.

```
DIMENSION X(2),Y(2)
```

```
A = 75.
```

```
B = 75.
```

```
X(1) = 25.
```

```
Y(1) = 75.
```

```
X(2) = 75.
```

```
Y(2) = 25.
```

COMMENTS

CALL USTART	'A'
CALL UOUTLN	'B'
CALL UMOVE (25.,25.)	'C'
CALL UPEN (A,B)	'D'
CALL UMOVE (X(1),Y(1))	'E'
CALL USET ('DASH')	'F'
CALL UPEN (X(2),Y(2))	
CALL UEND	'G'
STOP	
END	

#### COMMENTS:

- A: The first GCS call must be to USTART, a subroutine to initialize the status of the system.
- B: UOUTLN will be explained in Chapter III.
- C: All GCS subroutines begin with the letter 'U', user oriented, and have easy to remember names.
- D: Simple pen movements, visible UPEN's and invisible UMOVE's, are dependent upon the status of the system for their resulting actions.
- E: All arguments to GCS subroutine calls are either *real* or *character*.
- F: The status of the system can be modified, as in this case with USET, in order to obtain a wide range of output types.
- G: The last GCS call must be to UEND, a subroutine which performs any termination activity.

#### GCS CONVENTIONS

Easy to remember English-language descriptions rather than arbitrary codes are used to specify modes. For example, if one is in the preset condition and wishes to draw in polar coordinates with angles measured in radians and lines drawn as vectors (solid lines with arrowheads), one would merely use the following *user mode-setting* commands:

```
CALL USET ('POLAR')
CALL USET ('RADIAN')
CALL USET ('VECTOR')
```

To revert to drawing in rectangular coordinates with plain lines, the commands would be:

```
CALL USET ('RECTANGULAR')
CALL USET ('LINE')
```

Furthermore, there are those modes which require an additional parameter to be associated with that particular mode. For example, the preset polynomial degree to which a set of data can be fitted (using the subroutine ULSTSQ) is 5. If the user would prefer that his data be fitted to a polynomial of degree 3, he would use the *User Parameter Setting* command.

```
CALL UPSET ('POLYNOMIAL DEGREE',3)
```

(before the call to ULSTSQ). To re-establish the original polynomial of degree 5, the appropriate call would be

## CALL UPSET ('POLYNOMIAL DEGREE',5.)

The user need never be concerned with these modes of parameters unless the standard values do not suit him. He can easily change them to values he wants. Then he can forget all about them until he once again feels a need to change, or until the GSA is reinitialized.

The underlying principle which permits the use of this option setting technique is the fact that the GSA is in fact a table containing all of the options necessary to define an environment in which graphics activities are performed. These elements are preset to values which reflect the most commonly performed graphics operations. In order to make a particular graphics option, only one element of the table is altered.

*It will be normal practice in this manual to spell out completely the USET or UPSET option-names in full even if the name is very long, such as 'DOUBLEARROW' or 'DASHEDLINE'. Doing this is good form and good self-documentation for programs. GCS will, however, analyze only the first 4 characters in any option-name designator. If misspellings, truncations or non-standard wording occurs, the corrupted version will be fully acceptable to GCS provided that the first 4 characters of the character string are accurate.*

*With this system of organization, the beginning user—or the infrequent user—is insulated from the adverse effects which would tend to overwhelm the beginner in a system designed to provide flexibility for advanced and sophisticated users.*

*The user need be aware of, and pay attention to, only those degrees of freedom which he specifically wants to exploit in writing his program. GCS does not burden the user with long, complicated calling-parameter lists and complicated arbitrary codes. It can use short, easy-to-remember and easy-to-use commands convenient to both unsophisticated and sophisticated users.*

### Information in This Manual and Information Excluded From It

Some idea of the scope and range of the subroutines and options most likely to be of interest to the beginning user may be found in Appendices 'A' and 'B'. Sophisticated subroutines and options such as those involved in dealing with axis transformations, storing, manipulating and editing pictorial data structures have been completely omitted from this manual. Only a very limited subset of the most important and most frequently used subroutines and options from these appendices will actually be described in detail and discussed in this manual. For more complete programming information, see the section entitled GCS routines.



## **CHAPTER II**

### **GCS PROGRAMMING FUNDAMENTALS**

#### **Initialization**

The first GCS statement in any program must be

#### **CALL USTART**

Its functions are many, varied and important. Some of them are:

- A. It prepares the graphic terminal for plotting. In doing so, it clears the screen (or, in the case of plotter, requests the attendant to place a fresh sheet of paper on the plotbed), places the beam (or pen) in a standard position—coordinates (0,0) at the lower left corner of the standard plotting area.
- B. It sets all mode and parameter options at their standard default values.

In the standard default condition, the system is set to use Cartesian, absolute coordinates and to draw solid lines in a large square area contained within the screen or plotting bed of the graphics device. It is preset to accept the values of  $0. < X < 100.$  and  $0. < Y < 100.$  Additional default settings by USTART will be mentioned later.

#### **Reinitialization**

At any time the initial default conditions can be restored by:

#### **CALL URESET**

#### **Termination**

It is necessary to perform various file and buffer termination activities upon exit from GCS by putting as the last GCS command of any program

#### **CALL UEND**

Use of this statement will insure that all GCS output is completed.

#### **Erasing or Starting Off Again With a Clean Sheet**

At various stages in running a program, it is often worthwhile to erase everything which has been drawn or printed on a CRT screen or to ask the attendant of a plotter to replace the old sheet on the plotter with a fresh clean one. The call for performance of this function is:

#### **CALL UERASE**

The pen position remains unchanged when erasure occurs.

#### **Interactive Versus Batch**

In a batch program there is no need to suspend execution in order to view multiple plots

prior to erasing or starting off again with a clean sheet. In an interactive program an alarm can be sounded to indicate a plot is done by:

**CALL UBELL**

Then to allow viewing of the plot use:

**CALL UPAUSE**

### **Simple Line-Drawing**

The basic command in GCS for drawing lines is

**CALL UPEN (X,Y)**

This routine moves the beam (or pen) from its present coordinate position (**X0,Y0**) to a designated new location (X,Y). Pen status variables in the Graphics Status Area (GSA) determine what kind of line (if any) is drawn as a result of this movement.

Until some sort of overt action is taken to change pen status, every UPEN movement will cause a solid line to be drawn. (CALL USTART sets pen status to 'LINE'.) CALL USTART also sets the initial beam (pen) position to (0,0). Example II-1 shows a simple series of CALL UPEN commands which will draw a square by 4 basic pen-movements.

### **Invisible Lines**

If one wanted to draw a similar square starting at some location other than the origin, one needs to be able to move the beam or pen invisibly, that is, without drawing any line. One way to do this is to use the UMOVE subroutine

**CALL UMOVE (X,Y)**

The effect is identical to UPEN except that no line of any kind is drawn as the beam or pen moves from the old location (**X0,Y0**) to the new location (X,Y). Example II-2 shows a program which draws a square offset from the origin in this way. In this example we have used subroutine UOUTLN to outline the square we are drawing within. (UOUTLN is further explained in Chapter III.) The default origin is at the lower left corner (0,0).

### **Using a Pen Status Mode Change to Draw Invisible Lines**

The same effect as a call to UMOVE can be achieved by changing pen status mode from its default condition to 'NOLINE' by means of a call to USET.

**CALL USET ('NOLINE')**

causes the system to again draw a line. Example II-3 shows a program which uses this approach to draw the same image as Example II-2.

Another subroutine that is considered to be a simple line drawing subroutine, capable, if desired, of producing an invisible line as in the example above. A call to

**CALL UPEN1 (X,Y,'NOLINE')**

enables the user to generate an invisible line to (X,Y). The mode will be established as indicated for only that 1 pen movement. Effectively, the above call is the same as the CALL UMOVE (X,Y), or of the CALL USET ('NOLINE'), CALL UPEN (X,Y), CALL USET

('LINE') sequence. It will be shown throughout this text, however, that the chosen mode can be one of many different line modes other than 'NOLINE'.

### **Pen Status Mode Changes Allow Many Kinds of Lines to be Drawn**

The UPEN subroutines of GCS can draw many other kinds of lines, including lines in various colors (if the plotting device permits), dashed lines, lines with tics along their length, and lines of all these types with various kinds of terminating characters or symbols. The number of permutations of line types and terminators to be used at the end of a line is very large. Some of the more important are demonstrated later in Chapter VI. However some of the basic options are considered below.

### **Alternate Colors**

If the graphical device being used has the ability to display lines of different colors, the user can request that any subsequent line be generated with one of seven colors, using CALL USET (OPTION), where OPTION in this case is 'WHITE,' 'BLACK,' 'RED,' 'GREEN,' 'YELLOW,' 'BLUE,' 'MAGENTA,' or 'CYAN.'

Many terminals such as the Tektronix do not have a multi-color capability and, as such use the default color BLACK. Others, including most pen plotters, allow manual color change by replacement of the drawing pen by one of another color. A color mode change with such a device causes the plotter to position itself for pen replacement and a color change request to be typed out on the control device.

Most things done with colors can be done on devices which do not have a color capability by making distinctions through the use of plain lines, and various types of dashed lines, arrow lines, ticmarked lines and so forth.

### **Vectors or Lines with Arrowheads**

One frequently useful set of options is the arrowhead options. These are useful for drawing vectors, dimension lines, etc. These use the following pen status modes:

Pen Status	Description
'ARROW'	Draws a "forward vector" with its arrowhead at (X,Y)
'BACKARROW'	Draws "backward vector" with its arrowhead at (Xo,Yo)
'DOUBLEARROW'	Draws a "dimension line," i.e., a line with arrowhead pointing out at each end.

Example II-4 demonstrates the use of the 'ARROW' option.

### **Tic Marked Lines**

Another available option is tic lines, i.e., lines with tic marks at specified intervals along it. Pen status may be set to draw such lines by using a call to USET.

CALL USET ('TICLINE')

A call to UPSET

CALL UPSET ('TICINTERVAL',PARAMETER)



will change the interval along a ticline at which the tics will actually be placed. For example:

```
CALL UPSET ('TICINTERVAL',5.)
CALL UPEN1 (X,Y,'TICLINE')
```

will produce a ticline to (X,Y), marked off with tics at intervals of 5 units rather than the default value of 10. Example II-5 shows several ticline options.

### Dashed Lines

One can draw dashed lines by going to the 'DASHLINE' mode with a

```
CALL USET ('DASHLINE')
```

The length of the dashes and or the intervals between the dashes can be modified, if desired, with a

```
CALL UPSET ('SETDASH', DASHCODE)
```

the resulting dashed line will be repetitions of the effect caused by the combination of the digits (0-9) as follows:

DASHCODE Digit	Effect on Visible Units	Effect on Invisible Units
1	1	—
2	—	1
3	2	—
4	—	2
5	5	—
6	—	5
7	10	—
8	—	10
9	1 point	—
0	—	1 point

The specification of a single digit for DASHCODE will produce standard dashed (DASHCODE = 1-8) or standard dotted (DASHCODE = 9) lines. If the user desires a non-standard dashed line he must specify a DASHCODE of two or more digits, followed by a decimal point. The following sequence:

```
CALL USET ('DASH')
CALL UPSET ('SETDASH',76.)
CALL UPEN (X,Y)
```

will produce for example, a dashed line to (X,Y) consisting of 10 visible units, 5 invisible units, 10 visible units, 5 invisible units, etc. The physical length of a unit will be approximately 0.04 inches for the majority of the devices. Example II-6 illustrates several dashed line options. The number of possible types of dashed lines is limited only by the word length of the central computer.

### Polar Coordinates



The options absolute and rectangular (Cartesian) are established as defaults by USTART, because most drawings use this simple coordinate system. A polar coordinate system provides the user with the capability of referring to a location on the device in polar coordinates (R,THETA) rather than in rectangular coordinates (X,Y). Example II-7 illustrates use of polar plotting.

#### **Relative (Incremental) Plotting**

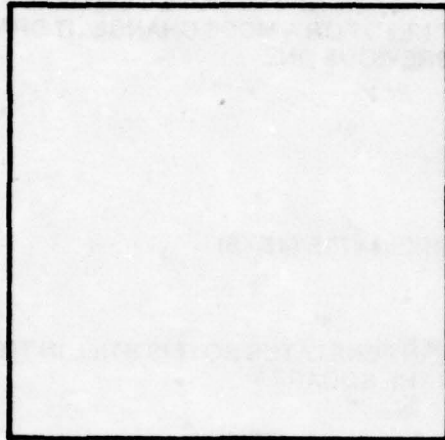
The relative coordinate mode interprets the coordinate pair of the pen request as units relative (positive or negative) to the previous pen position rather than as an absolute position. Example II-8 illustrates use of relative plotting.

```

C   THIS SAMPLE PROGRAM WILL DEMONSTRATE SIMPLE LINE-DRAWING BY
C   DRAWING A SQUARE BOX IN 4 PEN MOVEMENTS
C
C   INITIALIZE GCS
C   THIS INITIALIZATION SETS GCS TO RECTANGULAR, ABSOLUTE
C   COORDINATES, SOLID LINE PEN-DRAWING MODE, AND INITIAL PEN
C   COORDINATES (0,0)
C
C   CALL USTART
C
C   NOTE THAT ALL GCS SUBROUTINES (INCLUDING THE UPEN ROUTINE USE
C   TYPE REAL CALLING PARAMETERS. THUS COORDINATES MUST BE
C   ENTERED AS REAL NUMBERS, I.E. WITH DECIMAL POINTS
C
C   MOVE PEN TO (0,50) THEREBY DRAWING LINE (0,0) to (0,50)
C
C   CALL UPEN (0.,50.)
C
C   MOVE PEN TO (50,50) THEREBY DRAWING LINE (50,50) TO (50,0)
C
C   CALL UPEN (50.,0.)
C
C   MOVE PEN TO (0,0) THEREBY DRAWING LINE (50,0) TO (0,0)
C
C   CALL UPEN (0.,0.)
C
C   THIS COMPLETES DRAWING OF THE SQUARE
C
C   WRAP-UP. FIRST TERMINATE GCS BY CALL UEND. THEN STOP EXECUTION
C   WITH STOP. FINALLY END FORTRAN PROGRAM WITH END.
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE II-1



```
CALL USTART  
CALL UPEN (C8.,58.)  
CALL UPEN (C8.,58.)  
CALL UPEN (C8.,8.)  
CALL UPEN (C8.,8.)  
CALL UEND  
STOP  
END
```

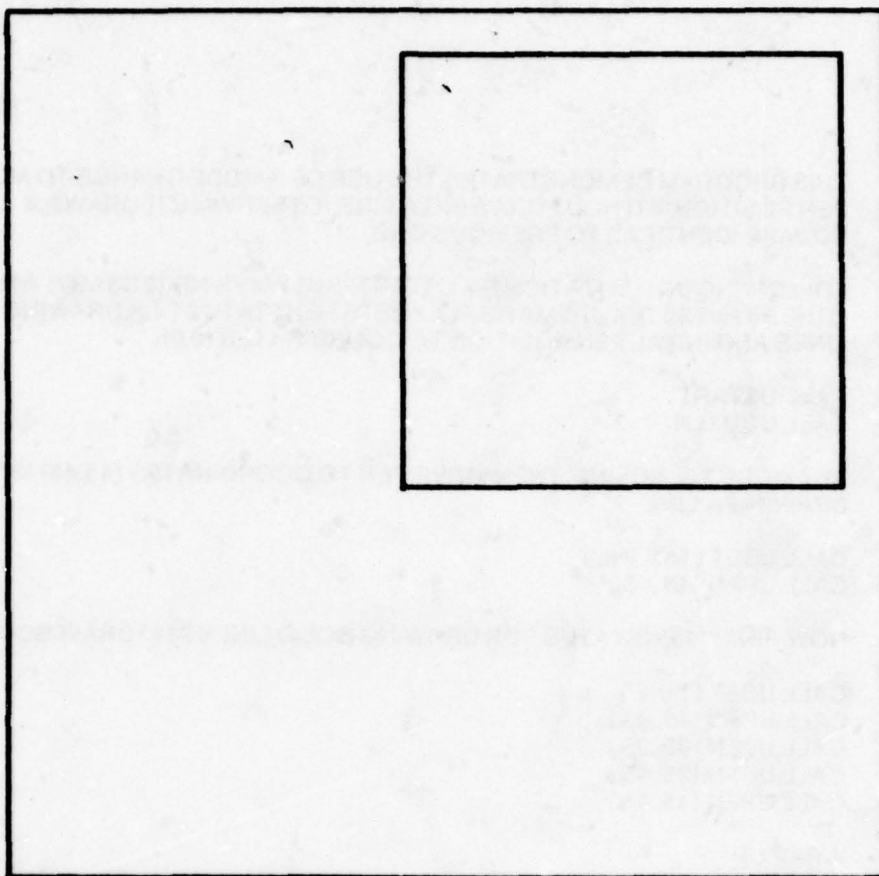
```

C   THIS PROGRAM DEMONSTRATES USE OF THE MOVE COMMAND TO MOVE
C   THE PEN INVISIBLY WITHOUT NEED FOR A MODE CHANGE. IT DRAWS A
C   SQUARE IDENTICAL TO THE PREVIOUS ONE.
C   INITIALIZE
C   CALL USTART
C   CALL UOUTLN
C   MOVE PEN INVISIBLY TO COORDINATES (45,45)
C   CALL UMOVE (45.,45.)
C   NO CHANGE HAS BEEN MADE IN PENSTATUS SO IT IS STILL IN THE DEFAULT
C   CASE OF SOLID LINES. DRAW THE SQUARE.
C   CALL UPEN (45.,95.)
C   CALL UPEN (95.,95.)
C   CALL UPEN (95.,45.)
C   CALL UPEN (45.,45.)
C   WRAP UP
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE II-2





```
CALL USTART
CALL UOUTLN
CALL UMOVE (45.,45.)
CALL UPIEN (45.,45.)
CALL UPIEN (85.,85.)
CALL UPIEN (85.,45.)
CALL UPIEN (45.,45.)
CALL UEND
STOP
END
```

```

U$START
U$OUTLN
U$MOVE
U$OPEN C
U$OPEN C
U$OPEN C
U$OPEN C
U$END

```

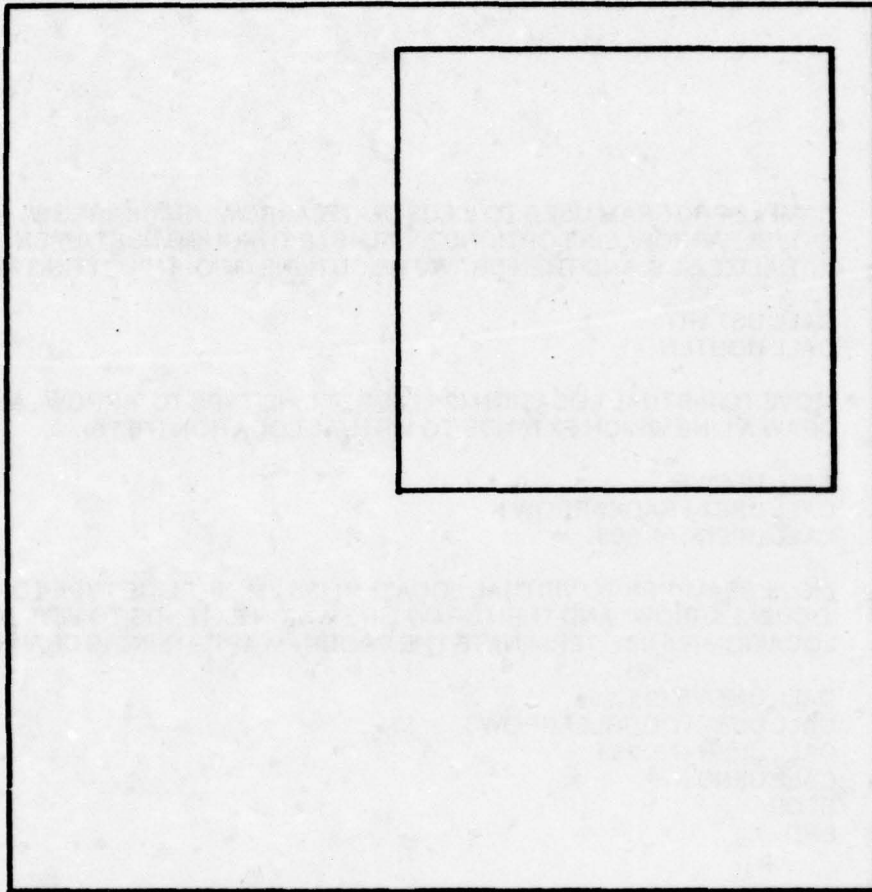
UNOVE (45., 45.)  
UPEN (45., 85.)  
UPEN (85., 85.)  
UPEN (85., 45.)  
UPEN (45., 45.)

```

C   THIS PROGRAM DEMONSTRATES THE USE OF A MODE CHANGE TO MOVE
C   PEN POSITION WITHOUT DRAWING A LINE. OTHERWISE IT DRAWS A
C   SQUARE IDENTICAL TO PREVIOUS ONE.
C
C   FOLLOWING INITIALIZATION BY USTART IS ALWAYS NECESSARY. AMONG
C   OTHER THINGS IT AUTOMATICALLY SETS PENSTATUS FOR DRAWING SOLID
C   LINES AND INITIAL PEN POSITION TO COORDINATES (0,0).
C
C   CALL USTART
C   CALL UOUTLN
C
C   SET MODE TO 'NOLINE' THEN MOVE PEN TO COORDINATES (45,45) WITHOUT
C   DRAWING A LINE
C
C   CALL USET ('NOLINE')
C   CALL UPEN (45.,45.)
C
C   NOW RESET PENSTATUS FOR DRAWING SOLID LINES AND DRAW SQUARE
C
C   CALL USET ('LINE')
C   CALL UPEN (45.,95.)
C   CALL UPEN (95.,95.)
C   CALL UPEN (95.,45.)
C   CALL UPEN (45.,45.)
C
C   WRAP UP
C
C   CALL UEND
C   STOP
C   END

```

### EXAMPLE II-3



CALL USTART  
CALL UCUTLN  
CALL USET ('NOLINE')  
CALL USET (45., 45.)  
CALL USET ('LINE')  
CALL USET (45., 85.)  
CALL USET (85., 85.)  
CALL USET (85., 45.)  
CALL USET (45., 45.)  
END



```

C   SAMPLE PROGRAM USED TO ILLUSTRATE 'ARROW', 'BACKARROW', AND
C   'DOUBLEARROW' LINE OPTIONS AVAILABLE THROUGH USET/UPEN.
C   INITIALIZE GCS, AND THEN DRAW THE OUTLINE OF OUR PLOTTING AREA

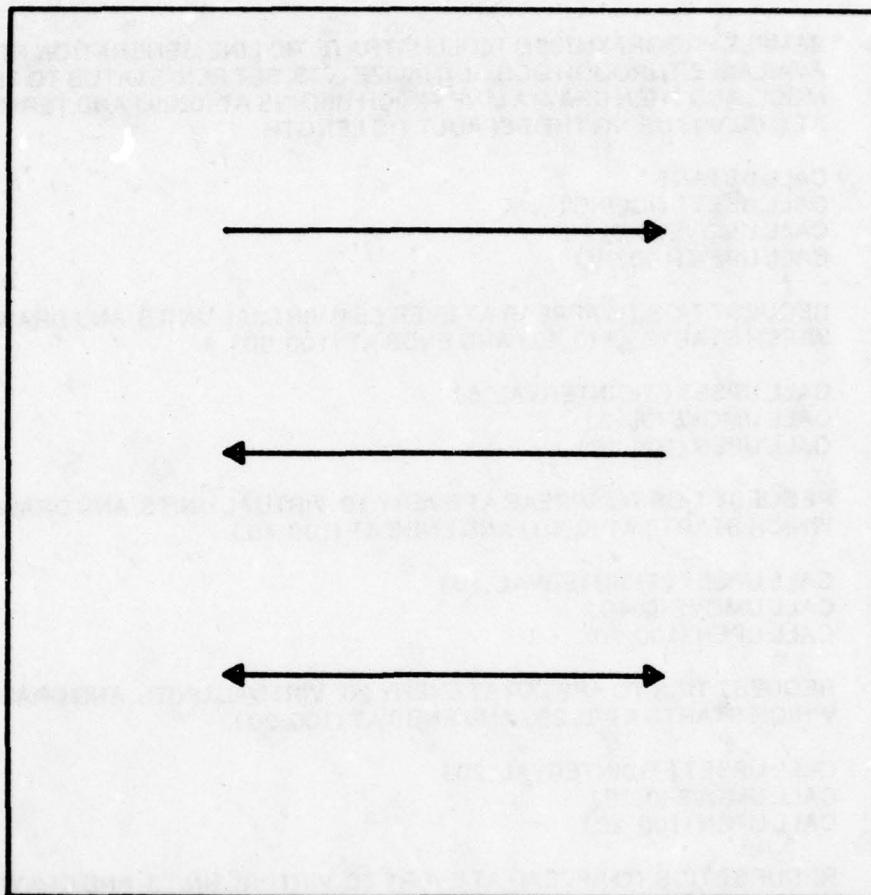
      CALL USTART
      CALL UOUTLN

C   MOVE TO VIRTUAL LOCATION (25,75), SET LINE TYPE TO 'ARROW', AND
C   DRAW A LINE WHICH EXTENDS TO VIRTUAL LOCATION (75,75).
C
      CALL UMOVE
      CALL USET ('BACKARROW')
      CALL UPEN (75.,50.)

C   MOVE BEAM/PEN TO VIRTUAL LOCATION (25,25), SET LINE TYPE TO
C   'DOUBLEARROW', AND THEN DRAW LINE WHICH EXTENDS TO VIRTUAL
C   LOCATION (75,25). TERMINATE THE PROGRAM AFTER LINE IS DRAWN.
C
      CALL UMOVE (25.,25.)
      CALL USET ('DOUBLEARROW')
      CALL UPEN (75.,25.)
      CALL UEND
      STOP
      END

```

#### EXAMPLE II-4



```
CALL USTART
CALL UOUTLN
CALL UMOVE (25.,75.)
CALL USET ("ARROW")
CALL UPEN (75.,75.)
CALL UMOVE (25.,50.)
CALL USET ("BACKARROW")
CALL UPEN (75.,50.)
CALL UMOVE (25.,25.)
CALL USET ("DOUBLEARROW")
CALL UPEN (75.,25.)
CALL UEND
STOP
END
```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE TIC LINE GENERATION PTIONS
C   AVAILABLE THROUGH GCS. INITIALIZE GCS, SET PEN-STATUS TO THE 'TIC'
C   MODE, AND THEN DRAW A LINE WHICH BEGINS AT (0.,99.) AND TERMINATES
C   AT (100.,99.) USING THE DEFAULT TIC LENGTH.

C   CALL USTART
C   CALL USET ('TICLINE')
C   CALL UMOVE (0.,99.)
C   CALL UPEN (100.,99.)

C   REQUEST TICS TO APPEAR AT EVERY 5.0 VIRTUAL UNITS, AND DRAW A LINE
C   WHICH STARTS AT (0.,60.) AND ENDS AT (100.,60.).

C   CALL UPSET ('TICINTERVAL',5.)
C   CALL UMOVE (0.,60.)
C   CALL UPEN (100.,60.)

C   REQUEST TICS TO APPEAR AT EVERY 10. VIRTUAL UNITS, AND DRAW A LINE
C   WHICH STARTS AT (0.,40.) AND ENDS AT (100.,40.).

C   CALL UPSET ('TICINTERVAL',10.)
C   CALL UMOVE (0.,40.)
C   CALL UPEN (100.,20.)

C   REQUEST TICS TO APPEAR AT EVERY 20. VIRTUAL UNITS, AND DRAW A LINE
C   WHICH STARTS AT (0.,20.) AND ENDS AT (100.,20.).

C   CALL UPSET ('TICINTERVAL',20.)
C   CALL UMOVE (0.,20.)
C   CALL UPEN (100.,20.)

C   REQUEST TICS TO APPEAR AT EVERY 50. VIRTUAL UNITS, AND DRAW A LINE
C   WHICH STARTS AT (0.,1.) AND ENDS AT (100.,1.).

C   CALL UPSET ('TICINTERVAL',50.)
C   CALL UMOVE (0.,1.)
C   CALL UPEN (100.,1.)
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE II-5



-----

-----

-----

-----

-----

-----

```
CALL USTART
CALL USET ("TICLINE")
CALL UMOVE(0.,00.)
CALL UPEN(100.,00.)
CALL UPSET ("TICINTERVAL",2.)
CALL UMOVE (0.,00.)
CALL UPEN (100.,00.)
CALL UPSET ("TICINTERVAL",5.)
CALL UMOVE (0.,00.)
CALL UPEN (100.,00.)
CALL UPSET ("TICINTERVAL",10.)
CALL UMOVE (0.,40.)
CALL UPEN (100.,40.)
CALL UPSET ("TICINTERVAL",20.)
CALL UMOVE (0.,20.)
CALL UPEN (100.,20.)
CALL UPSET ("TICINTERVAL",50.)
CALL UMOVE (0.,1.)
CALL UPEN (100.,1.)
CALL UEND
STOP
END
```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE DASHED LINE GENERATION
C   OPTIONS AVAILABLE THROUGH GCS. INITIALIZE GCS, SET THE PEN-STATUS
C   TO 'DASH' MODE, AND THEN DRAW A LINE WHICH BEGINS AT (0.,100.) AND
C   TERMINATES AT (100.,100.). THE DEFAULT VALUE OF DASH WILL BE USED
C   FOR THIS CASE.
C
C   CALL USTART
C   CALL USET ('DASHLINE')
C   CALL UMOVE (0.,100.)
C   CALL UPEN (100.,100.)
C
C   SET THE DASH SPECIFICATION TO 54., AND DRAW A LINE THAT STARTS AT
C   (0.,80.) AND ENDS AT (100.,80.).
C
C   CALL UPSET ('SETDASH',54.)
C   CALL UMOVE (0.,80.)
C   CALL UPEN (100.,80.)
C
C   SET THE DASH SPECIFICATION TO 5212., THEN GENERATE A LINE THAT
C   STARTS AT (0.,40.) AND ENDS AT (100.,40.).
C
C   CALL UPSET ('SETDASH',5212.)
C   CALL UMOVE (0.,40.)
C   CALL UPEN (100.,40.)
C
C   SET THE DASH SPECIFICATION TO 5434., THEN GENERATE A LINE THAT
C   STARTS AT (0.,20.) AND ENDS AT (100.,20.).
C
C   CALL UPSET ('SETDASH',5434.)
C   CALL UMOVE (0.,20.)
C   CALL UPEN (100.,20.)
C
C   SET THE DASH SPECIFICATION TO 96., AND DRAW A LINE THAT STARTS AT
C   (0.,0.) AND ENDS AT (100.,0.).
C
C   CALL UPSET ('SETDASH',96.)
C   CALL UMOVE (0.,0.)
C   CALL UPEN (100.,0.)
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE II-6

-----  
-----  
-----  
-----  
-----

.....

```
CALL USTART
CALL USET ("DASHLINE")
CALL UNOVE (8.,188.)
CALL UPEN (188.,188.)
CALL UPSET ("SETDASH",54.)
CALL UNOVE (8.,88.)
CALL UPEN (188.,88.)
CALL UPSET ("SETDASH",52.)
CALL UNOVE (8.,88.)
CALL UPEN (188.,88.)
CALL UPSET ("SETDASH",5212.)
CALL UNOVE (8.,48.)
CALL UPEN (188.,48.)
CALL UPSET ("SETDASH",5434.)
CALL UNOVE (8.,28.)
CALL UPEN (188.,28.)
CALL UPSET ("SETDASH",88.)
CALL UNOVE (8.,8.)
CALL UPEN (188.,8.)
CALL UEND
STOP
END
```

```

C      THIS PROGRAM WILL DEMONSTRATE THE USE OF POLAR PLOTTING BY
C      DRAWING A QUARTER OF A CIRCLE
C
C      INITIALIZE LINE DRAWING SPEED OF TERMNAL TO 120 CHARACTERS PER
C      SECOND
C
C      CALL USTART
C      CALL UPSET ('SPEED',120.)
C      CALL UERASE
C
C      INDICATE PLOTTING WILL BE IN POLAR COORDINATES (R,λ)
C
C      CALL USET ('POLAR')
C
C      PLOT A QUARTER OF A CIRCLE THAT HAS A RADIUS OF 100 IN NINE DEGREE
C      INCREMENTS FROM ZERO TO NINETY DEGREES
C
C      RADIUS= 100.
C      DO 100 I=1,11
C      ANGLE=FLOAT (I-1) * 9.
100    CALL UPEN (RADIUS,ANGLE)
C      CALL UPEN (0.,0.)
C
C      TERMINATION
C
C      CALL UEND
C      STOP
C      END

```

#### EXAMPLE II-7



```

C   THIS PROGRAM WILL DEMONSTRATE THE USE OF RELATIVE PLOTTING
C
C   INITIALIZE LINE DRAWING SPEED OF TERMINAL TO 120 CHARACTERS PER
C   SECOND
C
C   CALL USTART
C   CALL UPSET ('SPEED',120.)
C   CALL UERASE
C
C   INDICATE PLOTTING WILL BE DONE IN RELATIVE MODE AND DRAW A BOX
C   100 DATA UNITS ON A SIDE
C
C   CALL USET ('RELATIVE')
C   CALL UPEN (100.,0.)
C   CALL UPEN (0.,100.)
C   CALL UPEN (-100.,0.)
C   CALL UPEN (0.,-100.)
C
C   TERMINATION
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE II-8



## CHAPTER III

### VIRTUAL AND DEVICE GRAPHICS

#### Outlining the Plotting Area

To obtain an outline of the area within which all graphical activity will be performed, the programmer may invoke the GCS subroutine UOUTLN:

CALL UOUTLN

Although the use of UOUTLN certainly is not necessary for most graphical applications, there are many times (particularly during the learning and debugging phases) when it is useful for the programmer to know the physical boundaries of the display he is generating.

#### The Default Case and Simple Extensions

The examples presented so far have assumed the use of variables which fall into the range of values:

0 .LE. X .LE. 100

0 .LE. Y .LE. 100

with no specific control over where on the screen (or plotting bed) the pictures are actually drawn. This is the default case established by USTART.

The user can easily change the range of permissible values to the ranges

XMIN .LE. X .LE. XMAX

YMIN .LE. Y .LE. XMAX

by use of the command

CALL UWINDO (XMIN,XMAX,YMIN,YMAX)

This permits the user to plot anywhere in the space defined by the the four parameters in the subroutine call.

The values of the four parameters may be any FORTRAN real numbers permissible on the particular computer in use (subject only to the restriction XMIN .NE. XMAX and YMIN .NE. YMAX). See Example III-1.

#### Increasing and Decreasing Apparent Picture Size

Through the use of UWINDO, it is possible to increase or decrease the apparent picture size without having to change the values of the parameters required to generate the display. This facility effectively provides an elementary "zoom" capability which may be readily utilized under GCS. Example III-2 illustrates a simple six-step "zoom-out" by successively increasing the window specifications prior to generating each of the figures. Note that the pen commands required to draw the figure are identical in each of the six steps, and that the only dynamic entity is the window itself. Example III-3



illustrates a six-step "zoom-in" through decrementing the virtual window. It should be noted that an *increase* in the window specifications *decreases* the apparent picture size (zoom-out) and that a *decrease* of the window specifications yields an *increase* in apparent picture size (zoom-in).

### **Distortion**

It is important to note the ratio of the X-to-Y window specifications when defining new window boundaries, as ratios other than unity indicate that a distorted window will be constructed. Example III-4 illustrates this type of distortion by displaying the same basic information provided by Example III-3 with the exception that non-square windows are utilized.

### **Clipping the Edges of the Picture**

Should the graphics programmer attempt to display information at coordinate locations which lie outside of the window boundaries, GCS will "clip" the display at the window boundaries and any graphical activity attempted outside of the boundaries will not be displayed. Example III-4 illustrates the clipping feature applied to a simple six-level "zoom-in". We will again return to the topic of clipping after we first consider the concept of device coordinate addressing under GCS.

### **Expanding Flexibility and Capabilities: Device Mode**

When desired, GCS provides means of very detailed control of where pictures are drawn on the physical device. This may be done by going to the 'DEVICE' mode with a

CALL USET ('DEVICE')

Thereafter all coordinates are measured directly in device measurement units from the lower left corner of the display screen or plotting bed.

Several optional units of measure are available on the display screen or plotting bed:

CALL USET ('CENTIMETERS') causes horizontal and vertical distances to be measured in centimeters.

CALL USET ('FONTUNITS') causes horizontal measurement in units of the width of standard alphabetic characters (usually the hardware characters for the device being used) and vertical measurement in units of the height of the alphabetic characters.

CALL USET ('PERCENTUNITS') cause horizontal measurement in units of a percentage of the full width of the screen or plotting bed and vertical measurement in units of one percent of the full physical height of the screen or plotting bed.

CALL USET ('RASTERUNITS') causes horizontal and vertical measurement in terms of the smallest addressable unit for a particular device.

CALL USET ('INCHES') causes the horizontal and vertical measurement on the display to revert to the default case of inches.

See Example III-6

### **Windowing: A More Flexible and Advanced Concept of Picture Control**

Particular locations on the graphic device screen or plotting bed can be designated a device plotting area by the command

CALL UDAREA (XMIN,XMAX,YMIN,YMAX)

where XMIN,XMAX,YMIN and YMAX are measured in the currently defined units of measure in the device mode: inches, centimeters, percentunits, fontunits or rasterunits. Designation of this device plotting area has no effect in the device mode. However, in the virtual mode, either by default or by executing a call to USET ('VIRTUAL'), this device plotting area is at the heart of an extremely powerful concept of graphic control known as *windowing*. This concept is most easily visualized by referring to the next figure while following the description below and the sample programs and their output.

The user may pick any scale or range of values desired to plot by means of a call to UWINDO. For example, to plot values in the range  $0 < X < 1000$  and  $0 < Y < 1000$ , give the command

CALL UWINDO (0.,1000.,0.,1000.)

A call to UDAREA is used to specify where on the device to plot data. Then, plot in a 4-inch square at the left bottom of the display screen, first make sure that the unit of measure is 'INCHES' then give the command

CALL UDAREA (0.,4.,0.,4.)

Next, make sure that the mode is 'VIRTUAL' and do any plotting desired. Any values within the "virtual window" 0 .LE. X,Y .LE. 1000 is projected into the corresponding position of the "device plotting area" 0 .LE. X,Y, .LE. 4. For example, the user's numeric values of  $Y = 0$ ,  $X = 0$  would map into the lower left corner of the screen or plotting bed at position  $X = 0$ ,  $Y = 0$ . User values of  $X = 1000$ ,  $Y = 1000$  would map into the upper right corner of the 4-inch device plotting area at  $X = 4$ ,  $Y = 4$ . User values of  $X = 250$ ,  $Y = 750$  would correspondingly map into  $X = 1$ ,  $Y = 3$ .

But what happens if the user tries to draw a line to a point outside the 0 .LE. X, Y .LE. 1000 window, for example,  $X = 250$ ,  $Y = 1500$ ? Direct projection would indicate that such a line would rise vertically from the previous point to a point  $X = 1$ ,  $Y = 6$ , but such a line would pass out of the designated device plotting area. What happens is that the line is 'clipped' off at the edge of the window, i.e., drawn only to the limit of the window (which is, of course, also the limit of the device plotting area). Thus if any other lines or textual material had previously been put into adjacent area just outside the designated plotting area it would be protected against being overwritten by data which was considered to be irrelevant because it fell outside the expressed values of plotting interest specified by UWINDO. This 'clipping' action associated with mapping from a virtual window to a specified device area adds markedly to the power and ease of pictorial control in GCS.

Here are some examples of the use of windowing:

- A. **Convenient Units:** the imaginary or expandable plotting area or virtual window can be defined to correspond to any convenient units which the user would like to think in terms of.
- B. **Virtual Window Modification:** the user can selectively examine different areas of an entire display through the redefinition of the virtual window followed by the pen movements of that entire display. This idea can also be used to examine a small area of a display in great detail.

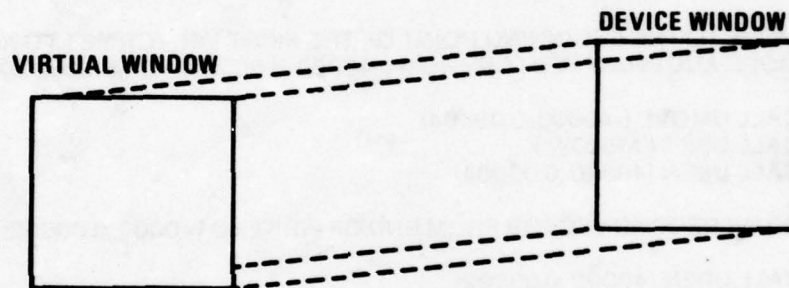
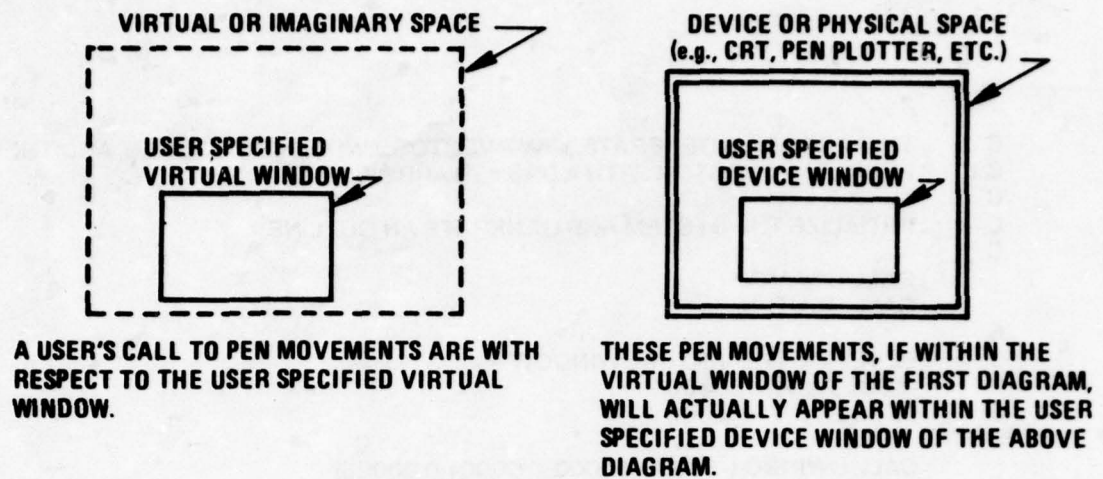
- C. **Device Window Modification:** the same display can be generated at many different locations on the device with the same pen movements by fixing the virtual window and changing the device window, as illustrated in Example III-7. An extension of this device window modification capability allows the user to achieve distortions, as in Example III-8.

#### **Device Independent Plotting**

The previous examples that used subroutine UDAREA were written for a Tektronix 4010/4013 terminal. A device independent program can be written by using 'percentunits' or

#### **CALL USTUD (ARRAY)**





THIS ACTUALLY OCCURS AS THE RESULT OF A MAPPING DIRECTLY FROM THE VIRTUAL WINDOW TO THE DEVICE WINDOW.

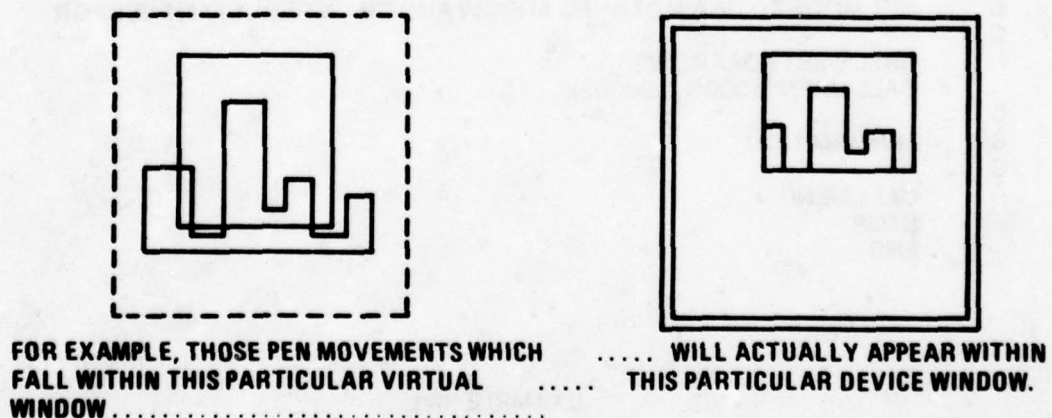


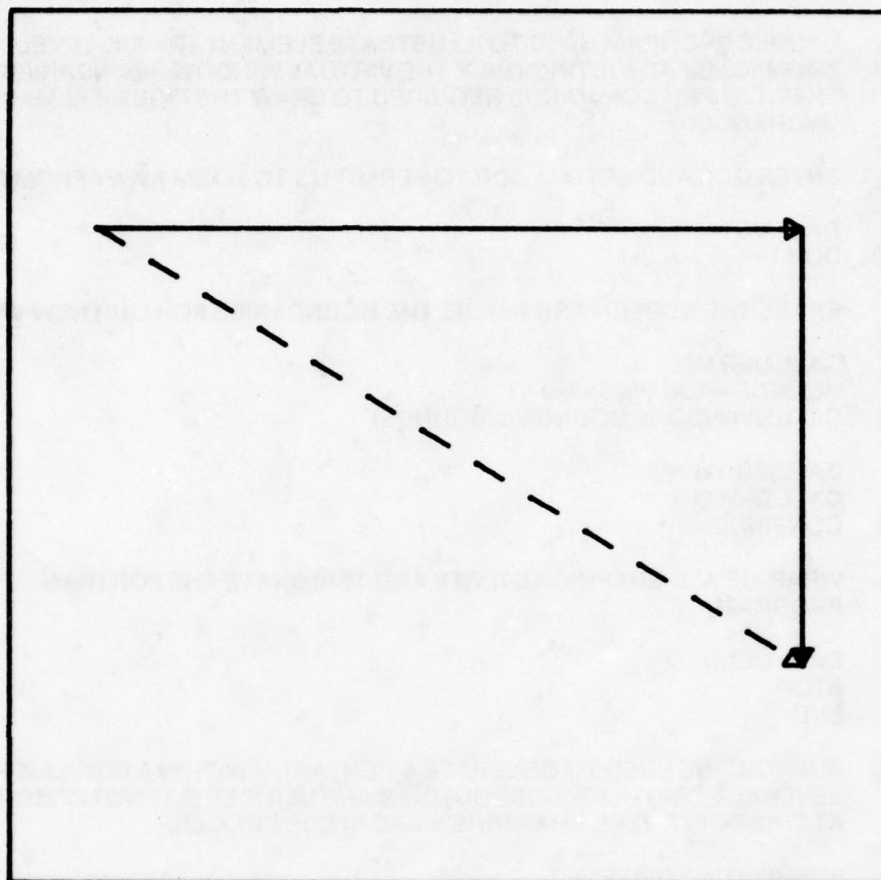
Figure 3-1. The Viewporting Concept

```

C   THIS PROGRAM GENERATES TWO VECTORS WITH ARROW LINES, AND THE
C   RESULTANT VECTOR WITH A DASHED ARROW LINE.
C
C   INITIALIZE THE SYSTEM AND GENERATE AN OUTLINE
C
C   CALL USTART
C   CALL UOUTLN
C
C   REDEFINE THE VIRTUAL WINDOW
C   -50000 < X < 50000
C   0.00001 < Y < 0.000
C
C   CALL UWINDO (-50000.,50000.,0.00001,0.00005)
C
C   DRAW THE TWO VECTORS
C
C   MOVE TO THE BEGINNING POINT OF THE FIRST VECTOR SET TO ARROW
C   MODE AND DRAW VECTOR FROM (-40000.,0.00004) TO (40000.,0.00004)
C
C   CALL UMOVE (-40000.,0.00004)
C   CALL USET ('ARROW')
C   CALL UPEN (40000.,0.00004)
C
C   DRAW SECOND VECTOR FROM END OF FIRST TO (40000.,0.00002)
C
C   CALL UPEN (40000.,0.00002)
C
C   MOVE TO BEGINNING OF VECTOR SYSTEM
C
C   CALL UMOVE (-40000.,0.00004)
C
C   SET MODE TO DRAW DASHED ARROW AND DRAW RESULTANT VECTOR
C
C   CALL USET ('DARROW')
C   CALL UPEN (40000.,0.00002)
C
C   TERMINATION
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE III-1



```

CALL USTART
CALL UOUTLN
CALL UNINDO (-50000.,50000.,8.88881,8.88885)
CALL UMOVE (-40000.,8.88884)
CALL USET ("ARROW")
CALL UPEN (40000.,8.88884)
CALL UPEN (40000.,8.88882)
CALL UMOVE (-40000.,8.88884)
CALL USET ("DARROW")
CALL UPEN (40000.,8.88882)
CALL UEND
STOP
END

```

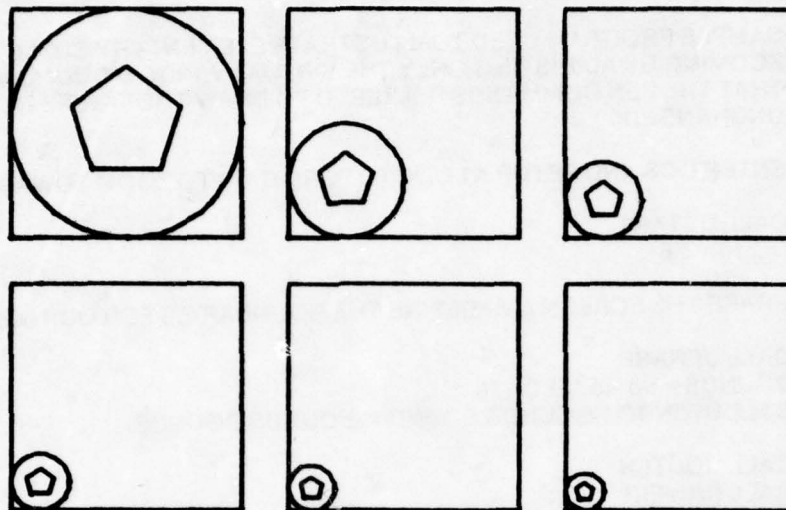


```

C   SAMPLE PROGRAM USED TO ILLUSTRATE ELEMENTARY SIX-LEVEL
C   ZOOMING BY ADJUSTING ONLY THE VIRTUAL WINDOW BOUNDARIES. NOTE
C   THAT THE PEN COMMANDS REQUIRED TO DRAW THE FIGURE REMAIN
C   UNCHANGED.
C   ENTER GCS AND SETUP LOOP TO PERMIT US TO ZOOM AWAY FROM FIGURE.
C
C   CALL USTART
C   DO 1 I=1,6
C
C   ERASE THE SCREEN AND DEFINE THE BOUNDARIES FOR OUR NEW WINDOW.
C
C   CALL UERASE
C   BOUNDS=100.*FLOAT(I)
C   CALL UWINDO (0.,BOUNDS,0.,BOUNDS)
C
C   CALL UOUTLN
C   CALL DRWFIG
C   1 CONTINUE
C
C   WRAP-UP ALL GRAPHIC ACTIVITY AND TERMINATE THE FORTRAN
C   PROGRAM.
C
C   CALL UEND
C   STOP
C   END
C
C   SUBROUTINE USED TO GENERATE A PENTAGON WITHIN A CIRCLE & PAUSE.
C   SEVERAL GCS UTILITY SUBROUTINES ARE UTILIZED BUT NOT DESCRIBED
C   AT THIS POINT. (SEE CHAPTERS V AND VI FOR DETAILS).
C
C   SUBROUTINE DRWFIG
C   CALL UCIRCLE (50.,50.,50.)
C   CALL UPLYGN (50.,50.,5.,25.)
C   CALL UBELL
C   CALL UPAUSE
C   RETURN
C   END

```

#### EXAMPLE III-2



```

CALL USTART
DO 1 I = 1, 6
CALL UERASE
BOUNDS = 100. * FLOAT(I)
CALL UMINO (8., BOUNDS, 8., BOUNDS)
CALL UOUTLN
CALL DRMFIG
1 CONTINUE
CALL UEND
STOP
END
SUBROUTINE DRMFIG
CALL UCIRCLE (50., 50., 50.)
CALL UPLYON (50., 50., 5., 25.)
CALL UELL
CALL UPAUSE
RETURN
END

```

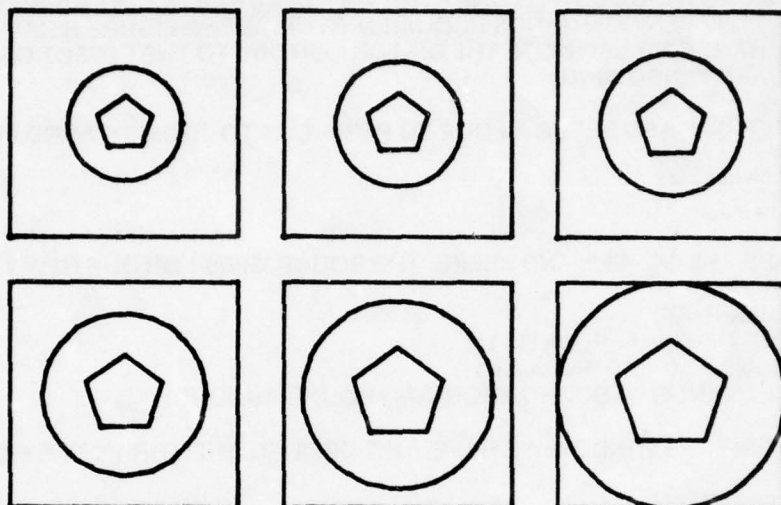
```

C      SAMPLE PROGRAM USED TO ILLUSTRATE ELEMENTARY SIX-LEVEL
C      ZOOMING BY ADJUSTING ONLY THE VIRTUAL WINDOW BOUNDARIES. NOTE
C      THAT THE PEN COMMANDS REQUIRED TO DRAW THE FIGURE REMAIN
C      UNCHANGED.
C
C      ENTER GCS AND SETUP A LOOP TO PERMIT US TO ZOOM TOWARD FIGURE.
C
C      CALL USTART
C      DO 1 I=1,6
C
C      ERASE THE SCREEN AND DEFINE THE BOUNDARIES FOR OUR NEW WINDOW.
C
C      CALL UERASE
C      BOUNDS=50.-(5.*FLOAT(I-1))
C      CALL UWINDO (-BOUNDS,BOUNDS,-BOUNDS,BOUNDS)
C
C      CALL UOUTLN
C      CALL DRWFIG
C      1  CONTINUE
C
C      WRAP-UP ALL GRAPHIC ACTIVITY AND TERMINATE THE FORTRAN
C      PROGRAM.
C
C      CALL UEND
C      STOP
C      END
C
C      SUBROUTINE USED TO GENERATE A PENTAGON WITHIN A CIRCLE & PAUSE.
C      SEVERAL GCS UTILITY SUBROUTINES ARE UTILIZED BUT NOT DESCRIBED
C      AT THIS POINT. (SEE CHAPTERS V AND VI FOR DETAILS).
C
C      SUBROUTINE DRWFIG
C      CALL UCRCLE (0.,0.,25.)
C      CALL UPLYGN (0.,0.,5.,12.5)
C      CALL UBELL
C      CALL UPAUSE
C      RETURN
C      END

```

### EXAMPLE III-3





```

CALL USTART
DO I = 1, 6
CALL UERASE
BOUNDS = 50. - (5. * FLOAT(I-1))
CALL UWINDO (-BOUNDS,BOUNDS,-BOUNDS,BOUNDS)
CALL UOUTLN
CALL DRWFIG
I CONTINUE
CALL UEND
STOP
END
SUBROUTINE DRWFIG
CALL UCIRCLE (0.,0.,25.)
CALL UPLYGN (0.,0.,5.,12.5)
CALL UELL
CALL UPAUSE
RETURN
END

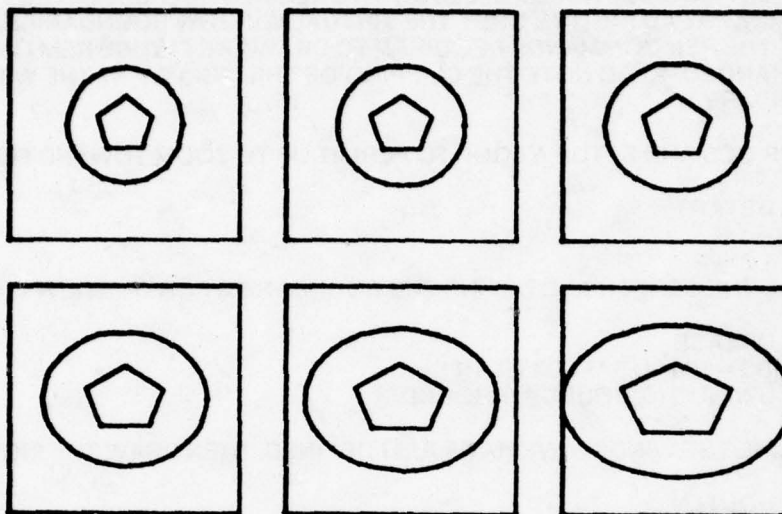
```

```

C      SAMPLE PROGRAM USED TO ILLUSTRATE ELEMENTARY SIX-LEVEL
C      ZOOMING BY ADJUSTING ONLY THE VIRTUAL WINDOW BOUNDARIES. NOTE
C      THAT THE PEN COMMANDS REQUIRED TO DRAW THE FIGURE REMAIN
C      UNCHANGED. ALSO NOTE THE DISTORTION DUE TO THE EFFECT OF NON-
C      SQUARE WINDOWING.
C      ENTER GCS AND SETUP A LOOP TO PERMIT US TO ZOOM TOWARD FIGURE.
C
C      CALL USTART
C      DO 1 I=1,6
C
C      ERASE THE SCREEN AND DEFINE THE BOUNDARIES FOR OUR NEW WINDOW
C
C      CALL UERASE
C      XBOUND=50.-(5.*FLOAT(I-1))
C      YBOUND=50.-(2.5*FLOAT(I-1))
C      CALL UWINDO (-XBOUND,XBOUND,-YBOUND,YBOUND)
C
C      OUTLINE THE WINDOW WE HAVE JUST DEFINED, THEN DRAW THE FIGURE.
C
C      CALL UOUTLN
C      CALL DRWFIG
C      CONTINUE
C 1     WRAP-UP ALL GRAPHIC ACTIVITY AND TERMINATE THE FORTRAN PROGRAM
C
C      CALL UEND
C      STOP
C      END
C
C      SUBROUTINE USED TO GENERATE A PENTAGON WITHIN A CIRCLE & PAUSE.
C      SEVERAL GCS UTILITY SUBROUTINES ARE UTILIZED BUT NOT DESCRIBED
C      AT THIS POINT. (SEE CHAPTERS V AND VI FOR DETAILS).
C
C      SUBROUTINE DRWFIG
C      CALL UCRGLE (0.,0.,25.)
C      CALL UPLYGN (0.,0.,5.,12.5)
C      CALL UBELL
C      CALL UPAUSE
C      RETURN
C      END

```

#### EXAMPLE III-4



```

CALL USTART
DO 1 I = 1, 6
CALL UERASE
XBOUND = 50. - (5. * FLOAT(I-1))
YBOUND = 50. - (2.5 * FLOAT(I-1))
CALL UINDO (-XBOUND, XBOUND, -YBOUND, YBOUND)
CALL UOUTLN
CALL DRWFIS
1 CONTINUE
CALL UEND
STOP
END
SUBROUTINE DRWFIS
CALL UCIRCLE (0.,0.,25.)
CALL UPLYN (0.,0.,5.,12.5)
CALL UBELL
CALL UPAUSE
RETURN
END

```

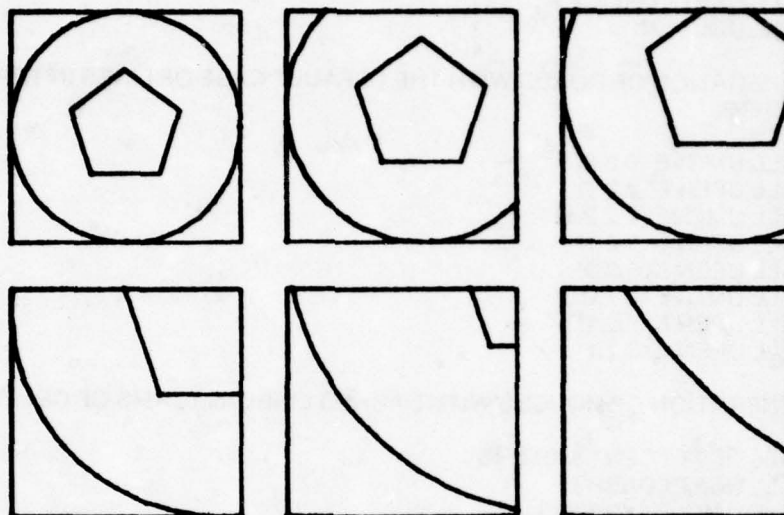


```

C      SAMPLE PROGRAM USED TO ILLUSTRATE ELEMENTARY SIX-LEVEL
C      ZOOMING BY ADJUSTING ONLY THE VIRTUAL WINDOW BOUNDARIES. NOTE
C      THAT THE PEN COMMANDS REQUIRED TO DRAW THE FIGURE REMAIN
C      UNCHANGED. ALSO NOTE THE CLIPPING OF THE DISPLAY AT THE WINDOW
C      BOUNDARY.
C
C      ENTER GCS AND SETUP A LOOP TO PERMIT US TO ZOOM TOWARD FIGURE.
C
C      CALL USTART
C      DO 1 I=1,6
C
C      ERASE THE SCREEN AND DEFINE THE BOUNDARIES FOR OUR NEW WINDOW.
C
C      CALL UREASE
C      BOUNDS=100.-(15.*FLOAT(I-1))
C      CALL UWINDO (0.,BOUNDS,0.,BOUNDS)
C
C      OUTLINE THE WINDOW WE HAVE JUST DEFINED, THEN DRAW THE FIGURE.
C
C      CALL UOUTLN
C      CALL DRWFIG
C      1 CONTINUE
C
C      WRAP-UP ALL GRAPHIC ACTIVITY AND TERMINATE THE FORTRAN
C      PROGRAM.
C
C      CALL UEND
C      STOP
C      END
C
C      SUBROUTINE USED TO GENERATE A PENTAGON WITHIN A CIRCLE AND
C      PAUSE. SEVERAL GCS UTILITY SUBROUTINES ARE UTILIZED BUT NOT
C      DESCRIBED AT THIS POINT. (SEE CHAPTERS V AND VI FOR DETAILS).
C
C      SUBROUTINE DRWFIG
C      CALL UCIRCLE (50.,50.,50.)
C      CALL UPLYGN (50.,50.,5.,25.)
C      CALL UBELL
C      CALL UPAUSE
C      RETURN
C      END

```

#### EXAMPLE III-5



```

CALL USTART
DO 1 I = 1, 6
CALL UERASE
BOUNDS = 100. - (15. * FLOAT(I-1))
CALL UWINDO (8., BOUNDS, 8., BOUNDS)
CALL UOUTLN
CALL DRWFIS
1 CONTINUE
CALL UEND
STOP
END
SUBROUTINE DRWFIS
CALL UCIRCLE (50., 50., 50.)
CALL UPLYON (50., 50., 5., 25.)
CALL UELL
CALL UPAUSE
RETURN
END

```

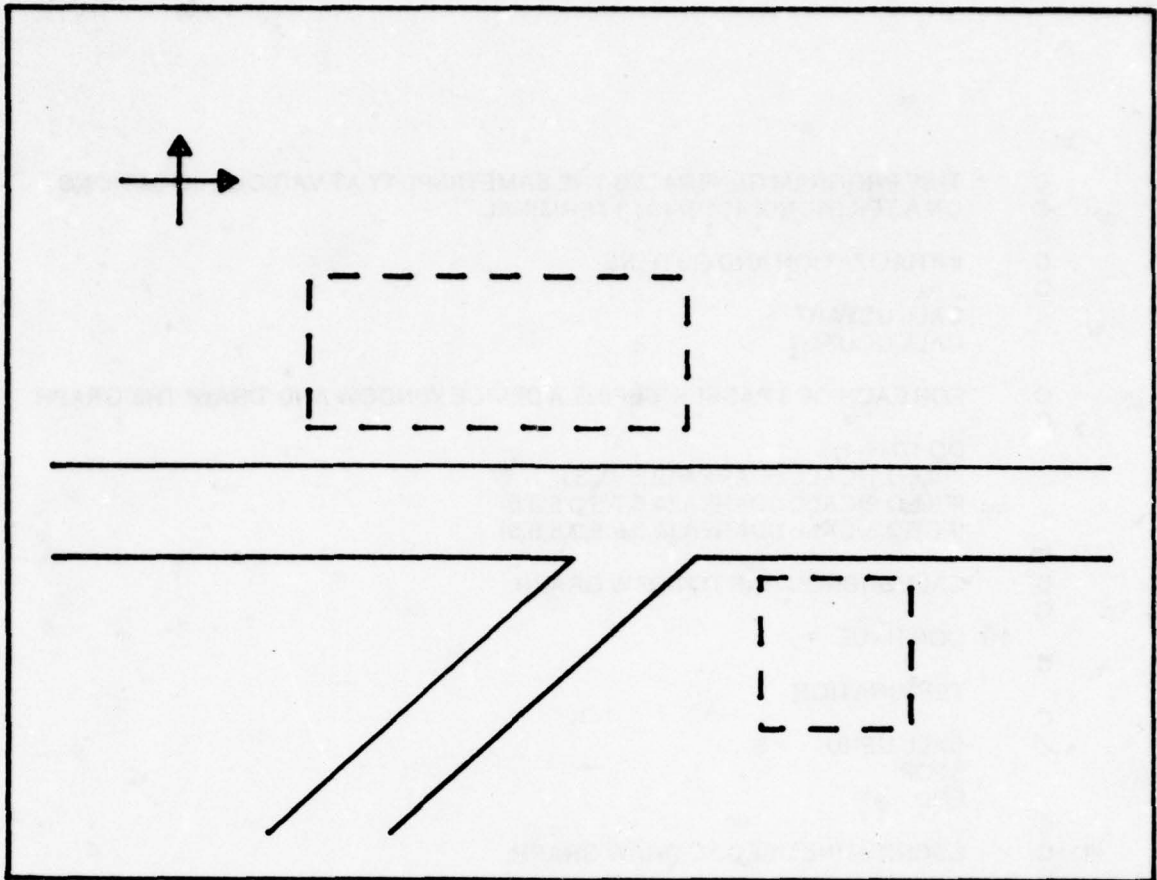
```

C   THIS PROGRAM GENERATES A SIMPLE STREET DIAGRAM FOR A TEKTRONIX
C   4010/4013 TERMINAL.
C
C   INITIALIZATION, DEVICE MODE ENTRY, AND OUTLINE GENERATION
C
C   CALL USTART
C   CALL USET ('DEVICE')
C   CALL UOUTLN
C
C   GENERATION OF ROADS WITH THE DEFAULT CASE OF LINES IN TERMS OF
C   INCHES.
C
C   CALL UMOVE (0.3,2.7)
C   CALL UPEN (7.2,2.7)
C   CALL UMOVE (7.2,2.1)
C   CALL UPEN (4.5,2.1)
C   CALL UPEN (2.5,0.3)
C   CALL UMOVE (1.7,0.3)
C   CALL UPEN (3.7,2.1)
C   CALL UPEN (0.3,2.1)
C
C   GENERATION OF HOUSES WITH DASHED LINES IN TERMS OF CENTIMETERS.
C
C   CALL USET ('CENTIMETERS')
C   CALL USET ('DASH')
C   CALL UMOVE (12.5,5.0)
C   CALL UPEN (15.5)
C   CALL UPEN (15.2.5)
C   CALL UPEN (12.5.2.5)
C   CALL UPEN (12.5,5)
C   CALL UMOVE (5.,7.5)
C   CALL UPEN (11.3,7.5)
C   CALL UPEN (11.3,10.0)
C   CALL UPEN (5.,10.)
C   CALL UPEN (5.,7.5)
C
C   GENERATION OF DIRECTION REFERENCES WITH ARROW LINES IN TERMS OF
C   PERCENTUNITS
C
C   CALL USET ('PERCENTUNITS')
C   CALL USET ('ARROW')
C   CALL UMOVE (10.,80)
C   CALL UPEN (20.,80.)
C   CALL UMOVE (15.,75.)
C   CALL UPEN (15.,85.)
C
C   TERMINATION
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE III-6





```

CALL USTART
CALL USET ('DEVICE')
CALL UOUTLN
CALL UMOVE (8.3,2.7)
CALL UPEN (7.2,2.7)
CALL UMOVE (7.2,2.1)
CALL UPEN (4.5,2.1)
CALL UPEN (2.5,8.3)
CALL UMOVE (1.7,8.3)
CALL UPEN (3.7,2.1)
CALL UPEN (8.3,2.1)
CALL USET ('CENTIMETERS')
CALL USET ('DASH')
CALL UMOVE (12.5,5.8)
CALL UPEN (15.,5.)
CALL UPEN (15.,2.5)
CALL UPEN (12.5,2.5)
CALL UPEN (12.5,5.)
CALL UMOVE (5.,7.5)
CALL UPEN (11.3,7.5)
CALL UPEN (11.3,18.8)
CALL UPEN (5.,18.)
CALL UPEN (5.,7.5)
CALL USET ('PERCENTUNITS')
CALL USET ('ARROW')
CALL UMOVE (18.,88.)
CALL UPEN (28.,88.)
CALL UMOVE (15.,75.)
CALL UPEN (15.,85.)
CALL UEND
STOP
END

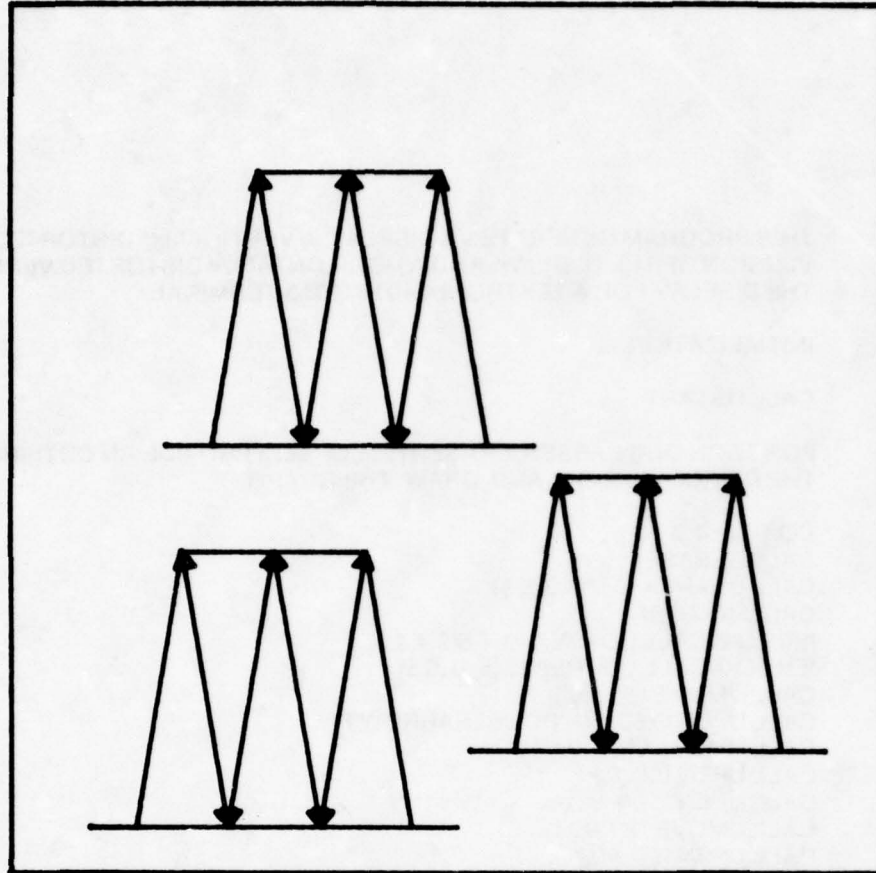
```

```

C   THIS PROGRAM GENERATES THE SAME DISPLAY AT VARIOUS LOCATIONS
C   ON A TEKTRONIX 4010/4013 TERMINAL.
C   INITIALIZATION AND OUTLINE.
C   CALL USTART
C   CALL UOUTLN
C   FOR EACH OF 3 PASSES, DEFINE A DEVICE WINDOW AND 'DRAW' THE GRAPH.
C   DO 10 I=1,3
C   IF(I.EQ.1)CALL UDAREA (2.,5.,0.,3.)
C   IF(I.EQ.2)CALL UDAREA (4.5,7.5,0.5,3.5)
C   IF(I.EQ.3)CALL UDAREA (2.5,5.5,2.5,5.5)
C   CALL SUBROUTINE TO DRAW GRAPH.
C   10 CONTINUE
C   TERMINATION
C   CALL UEND
C   STOP
C   END
C   SUBROUTINE USED TO DRAW GRAPH.
C   SUBROUTINE GRAFIT
C   CALL UMOVE (10.,10,10.)
C   CALL USET ('LINE')
C   CALL UPEN (90,10.)
C   CALL UMOVE (20.,10.)
C   CALL UPEN (30.,70.)
C   CALL UPEN (70.,70.)
C   CALL UPEN (80.,10.)
C   CALL UMOVE (30.,70.)
C   CALL USET ('DOUBLEARROW')
C   CALL UPEN (40.,10.)
C   CALL UPEN (50.,70.)
C   CALL UPEN (60.,10.)
C   CALL UPEN (70.,70.)
C   RETURN
C   END

```

#### EXAMPLE III-7



```

CALL USTART
CALL UDUTLN
DO 18 I = 1, 3
IF (I.EQ. 1) CALL UDAREA (2.,5.,8.,9.)
IF (I.EQ. 2) CALL UDAREA (4.5,7.4,8.5,9.5)
IF (I.EQ. 3) CALL UDAREA (2.5,5.5,2.5,5.5)
CALL GRAFIT
18 CONTINUE
CALL UEND
STOP
END
SUBROUTINE GRAFIT
CALL UMOVE (18.,18.)
CALL USET ('LINE')
CALL UPEN (68.,18.)
CALL UMOVE (28.,18.)
CALL UPEN (38.,78.)
CALL UPEN (78.,78.)
CALL UPEN (68.,18.)
CALL UMOVE (38.,78.)
CALL USET ('DOUBLEARROW')
CALL UPEN (48.,18.)
CALL UPEN (58.,78.)
CALL UPEN (68.,18.)
CALL UPEN (78.,78.)
RETURN
END

```

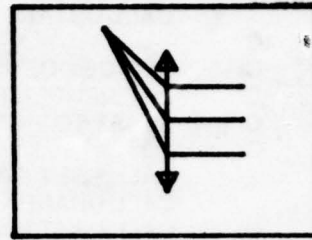
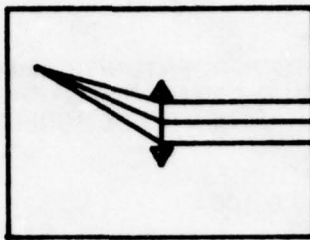
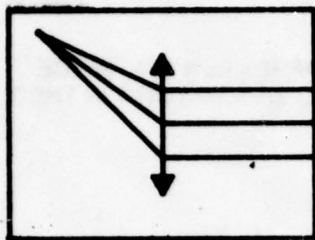


```

C   THIS PROGRAM GENERATES A DISPLAY, A VERTICALLY DISTORTED
C   VERSION OF THE DISPLAY, AND A HORIZONTALLY DISTORTED VERSION OF
C   THE DISPLAY FOR A TEKTRONIX 4010/4013 TERMINAL.
C
C   INITIALIZATION.
C
C   CALL USTART
C
C   FOR EACH OF 3 PASSES, ERASE THE SCREEN, PROVIDE AN OUTLINE, DEFINE
C   THE DEVICE WINDOW, AND 'DRAW' THE GRAPH.
C
C   DO 10 I=1,3
C   CALL UERASE
C   CALL UDAREA (0.,7.5,0.,5.5)
C   CALL UOUTLN
C   IF(I.EQ.2)CALL UDAREA (0.,7.5,2.4,4.)
C   IF(I.EQ.3)CALL UDAREA (3.,5.,0.,5.5)
C   CALL UMOVE (50.,20.)
C   CALL UPEN1 (50.,80.,'DOUBLEARROW')
C   CALL UMOVE (10.,90.)
C   CALL UPEN (50.,65.)
C   CALL UPEN (100.,65)
C   CALL UMOVE (10.,90.)
C   CALL UPEN (50.,50.)
C   CALL UPEN (100.,50.)
C   CALL UMOVE (10.,90.)
C   CALL UPEN (50.,35.)
C   CALL UPEN (100.,35.)
10  CONTINUE
C
C   TERMINATION.
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE III-8



```

CALL USTART
DO 10 I = 1, 3
CALL UERASE
CALL UDAREA (9.,7.4,9.,5.5)
CALL UOUTLN
IF (I.EQ. 2) CALL UDAREA (9.,7.4,2.,4.)
IF (I.EQ. 3) CALL UDAREA (9.,5.,9.,5.5)
CALL UMOVE (50.,20.)
CALL UPEN (50.,80., 'DOUBLEARROW')
CALL UMOVE (10.,80.)
CALL UPEN (50.,85.)
CALL UPEN (100.,85.)
CALL UMOVE (10.,80.)
CALL UPEN (50.,50.)
CALL UPEN (100.,50.)
CALL UMOVE (10.,80.)
CALL UPEN (50.,95.)
CALL UPEN (100.,95.)
10 CONTINUE
CALL UEND
STOP
END

```

```

C   THIS PROGRAM DEMONSTRATES HOW TO SUPPORT DEVICE
C   INDEPENDENCE.
C
C   DIMENSION ARRAY (8)
C
C   INITIALIZE
C
C   CALL USTART
C
C   SET 'DEVICE' UNITS TO 'PERCENTUNITS' and call UDAREA for 100%. SINCE
C   ONE PERCENTUNIT IN THE X MAY DIFFER FROM ONE PERCENTUNIT IN THE Y,
C   RESET TO 'INCHES' AND THEN DIVIDE SCREEN.
C
C   CALL USET ('PERCENTUNITS')
C   CALL UDAREA (0.,100.,0.,100.)
C   CALL USET ('INCHES')
C   CALL USTUD ('ARRAY')
C
C   ASSUME SCREEN IS LONGER IN X
C
C   XMIN = ARRAY(5)
C   XMAX = AMIN1 (ARRAY(6)/2.,ARRAY(8))
C   YMIN = ARRAY(7)
C   YMAX = XMAX
C
C   PLOT THE SAME FIGURE ON HALF OF THE SCREEN
C
C   CALL UDAREA (XMIN,XMAX,YMIN,YMAX)
C   CALL UOUTLN
C   CALL FIGURE
C   CALL UDAREA (XMAX,XMAX*2.,YMIN,YMAX)
C   CALL UOUTLN
C   CALL FIGURE
C
C   TERMINATION
C
C   CALL UEND
C   STOP
C   END
C
C   SUBROUTINE TO GENERATE A POLYGON
C
C   SUBROUTINE FIGURE
C   CALL UPLYGN (50.,50.,5.,25.)
C   RETURN
C   END

```

#### EXAMPLE III-9



## CHAPTER IV

### ALPHANUMERIC OUTPUT

#### Basic Alphanumeric Output

Under many circumstances, the graphics programmer would prefer to manipulate alphanumeric data with the same ease that pertains to strictly graphical information. Such a capability would expedite the labeling of figures, allow numerical values to be printed at a specific location on the display, and permit a discrete separation of text from graphical information should the user so desire. In order to provide these capabilities, GCS has incorporated two very powerful alphanumeric output and editing routines, UPRINT and UWRITE, which may be called by the following sequences:

```
CALL UPRINT (X,Y,DATA)
CALL UWRITE (X,Y,DATA)
```

X and Y are used to specify the starting position (in current units) of the first character of the output text. All positional and coordinate addressing related to characters is assumed to specify the position of the lower-left corner of the given character. DATA is interpreted in the following manner, based upon one of the four options available to the user through a call to USET before invoking UPRINT or UWRITE:

- A. **TEXT**—*Under this default option, the routines will assume that DATA contains Hollerith information delimited by a terminator and will output all text up to, but not including that delimiter, beginning at the location specified by (X,Y). CALL USET ('TERMINATOR', CHARACTER) can change the terminator from the default backslash character. There are four 'hardware' character sizes that can be changed by CALL USET(SIZE) where SIZE can be 'EXTRALARGE', 'LARGE', 'MEDIUM', or 'SMALL' (default).*
- B. **REALNUMBER**—*DATA is assumed to be a single-valued parameter whose contents is assumed to be of type REAL. Editing will take place using 'G' format, and the text will be printed beginning at (X,Y).*
- C. **INTEGER**—*As in the case of REALNUMBER, DATA is assumed to be single-valued and of type REAL. The routines will perform a REAL to INTEGER conversion and will print the result in 'I' format.*
- D. **XYCOORDINATES**—*DATA is assumed to be a 2 element REAL array whose values specify two numbers to be edited and printed in the form: (X,Y).*

UPRINT and UWRITE perform identical functions and differ only in the action taken upon return from execution of the given subroutine. Upon exit, UWRITE moves the beam position back to its original coordinates at the time it was invoked, whereas UPRINT leaves the beam position at the end of the output text. Example IV-1 illustrates the four options above.

Alphanumeric output through GCS is perhaps most conveniently manipulated through the use of FONTUNIT coordinates under DEVICE mode. Under this USET option, the (X,Y) coordinates are expressed directly in number of characters horizontally or vertically.

#### Margining

Since alphanumeric output through UPRINT and UWRITE is unaffected by the virtual

window (when in VIRTUAL mode), the user is provided the capability to horizontally and vertically margin his output (when in device mode) through an invocation to UMARGN:

CALL UMARGN (XLEFT,XRIGHT,YBOTTM,YTOP)

XLEFT and XRIGHT represent the desired left-most and right-most horizontal positions expressed in DEVICE units within which alphanumeric output may be permitted. Maximum and minimum vertical margins are established by YBOTTM and YTOP respectively. Should values of XLEFT, XRIGHT, YTOP, and YBOTTM be specified such that  $XRIGHT < XLEFT$  or  $YTOP < YBOTTM$  UMARGN will ignore the specified setting and the margins will reflect their values prior to the call to UMARGN.

Margining affects alphanumeric output (when in device mode) in the following manner:

- A. Should any character extend beyond the right margin, an automatic carriage return and line feed are generated, followed by a position to the left margin and output resumed.
- B. Should the user position the beam at a location to the left of the left margin, the beam is positioned to the left margin before output is initiated.
- C. Should output be attempted at vertical positions outside the closed interval defined by the minimum and maximum vertical margins, the beam is moved to the maximum vertical margin before any output is begun.

See Example IV-2 for an example of margining.

#### **Bulk or Mixed Alphanumeric Output**

For applications requiring bulk or mixed alphanumeric output, the graphics programmer should consider the use of subroutine UPRNT1 and UWRIT1, which may be invoked by the following calling sequence:

CALL UPRNT1 (DATA,OPTION)  
CALL UWRIT1 (DATA,OPTION)

OPTION is a single-valued character variable which specifies the format or mode under which DATA is to be edited and displayed; i.e., TEXT, REALNUMBER, GREEN, etc. DATA specifies the actual information which is to be output by UPRNT1 and UWRIT1 under the given OPTION. It should be noted that the constraints on DATA are identical to those imposed by UPRINT and UWRITE.

UPRNT1 and UWRIT1 differ from UPRINT and UWRITE in the following respects:

- A. Since no coordinate specifications are passed in the calling sequence, the alphanumeric output will begin at the current beam position upon entry to the subroutine.
- B. The effects of OPTION apply only to the current output operation and upon exit from the subroutine these effects are removed from the GSA.

#### **Fortran Input-Output**

For alphanumeric output used in interactive input-output communications and various control and coordination activities (such as the printout of warning or error notifications), the convenience of permitting the use of normal FORTRAN input-output statements may

override the advantages of using UPRINT or UWRITE to process text. For such activities to be performed successfully, it is essential that the terminal device be set for the receipt of alphanumeric rather than graphic information, a task which is accomplished through a call to **UALPHA**. On systems that buffer the graphics output, **UALPHA** must be called prior to the Fortran I/O to flush the graphics output. It should be noted that all alphanumeric FORTRAN output is unaffected by the margins established by the user.

### **Single Character Output**

In isolated instances, it is desirable to combine the output of graphical and alphanumeric information into one composite operation. This capability is provided through the 'CHARACTER' option available with UPEN, whereby a single printable character used as a parameter to USET identifies that parameter as the current character for printing. Thus, in order to print the character 'A' at (X,Y), the following commands would be specified:

```
CALL USET ('LA')  
CALL UPEN (X,Y)
```

Upon execution of the call to UPEN, the character 'A' will terminate the line drawn under the current line option. It should be noted that the CHARACTER mode of operation used in the above manner will remain in effect until any one of the acceptable UPEN suffixes is specified; e.g., CALL USET ('LNULL'). See Example IV-3.

An additional facility for single-character output is provided through subroutine UAOUT which is called in the following manner:

```
CALL UAOUT (CHAR)
```

CHAR is a single character expressed in Hollerith format either as a quoted character string or as an explicit variable. Since UAOUT outputs the given character at the current beam position, the user should ensure that the beam is positioned to the proper coordinates before invoking UAOUT. Character output through UAOUT is subject to the current margin constraints applicable to UPRINT and UWRITE: in addition, this output differs from that available through UPEN in that UAOUT positions the beam to next character position after the specified character has been printed, whereas the UPEN option always maintains the beam position at the center of the desired character. See Example IV-4 and compare to Example IV-3.

### **Software Character Output**

The preceding discussion of alphanumeric output has been applicable only to **HARDWARE** character generation. Under certain circumstances, however, the sophisticated user may desire character output which is rotated or scaled to suit a particular application. For examples see Example IV-5, and IV-6. This capability is provided in GCS through the **SOFTWARE** character option available through USET. Under this option, all output is subject to the constraints imposed by the virtual window, whenever UPRINT is called in 'VIRTUAL' space. Although **SOFTWARE** characters are manipulated using the same routines as those for output of **HARDWARE** characters, interested users are directed to the relevant subroutine writeups for a discussion of the limitations imposed on this form of output.



```

C   SAMPLE PROGRAM USED TO ILLUSTRATE OPTIONS AVAILABLE THROUGH
C   'UPRINT' AND 'UWRITE'. DEFAULT VIRTUAL WINDOW AND DEVICE AREA WILL
C   BE USED.
C
C   DEFAULT OPTION FOR 'UPRINT' AND 'UWRITE' IS FOR 'TEXT'. USE THIS
C   DEFAULT TO OUTPUT A LINE OF TEXT AT SAMPLE COORDINATE LOCATION
C   (0.,100.) NOTE THE SEMICOLAN(;) IS THE DELIMITER.
C
C   CALL USTART
C   CALL UPSET ('TERMINATOR',';')
C   CALL UPRINT (0.,100., 'THIS IS A SAMPLE LINE OF OUTPUT TEXT;')
C
C   SPECIFY THE 'REALNUMBER' OF OPERATION AND USE UWRITE TO PRINT AT
C   MIDSCREEN (50.,50.) A TYPICAL REAL NUMBER (100.)
C
C   CALL USET ('REALNUMBER')
C   CALL UWRITE (50.,50.,100.)
C
C   SPECIFY 'INTEGER' MODE AND PRINT AT COORDINATES (75.,25.) THE
C   SAMPLE INTEGER -123456789 NOTICE THAT SINCE ALL GCS PARAMETERS
C   ARE REAL NUMBERS, EVEN THIS INTEGER MUST BE PASSED IN REAL
C   NUMBER FORM I.E. -123456789.
C
C   CALL USET ('INTEGER')
C   CALL UPRINT (75.,25.,-123456789.)
C
C   EXAMPLE OF USING 'XYCOORDINATE' OPTION TO PRINT XYCOORDINATES.
C   NOTE THE VARIED FORM OF OUTPUT OF REAL NUMBERS WITH 'G' FORMAT
C
C   FIRST ESTABLISH A 2-COORDINATE ARRAY: COORD(2). PUT SAMPLE DATA
C   POINT (25.,0.99999999E19) IN ARRAY. NOTE THAT COORDINATES SPECIFIED
C   IN ARRAY COORD ARE PRINTED. THEY MAY OR MAY NOT BE SAME AS
C   COORDINATES WHICH SPECIFY WHERE PRINTING IS TO OCCUR.
C
C   DIMENSION COORD (2)
C   DATA COORD/25.,0.99999999E19/
C
C   CALL USET ('XYCOORDINATE')
C   CALL UWRITE (25.,75.,COORD)
C
C   END GCS (CALL UEND); STOP EXECUTION (STOP), END PROGRAM (END)
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE IV-1

THIS IS A SAMPLE LINE OF OUTPUT TEXT

(25...1888E+28)

188.

-123456789

```
DIMENSION COORD (2)
DATA COORD/25.,8.8888888E18/
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UPINT (8.,75., 'THIS IS A SAMPLE LINE OF OUTPUT TEXT,')
CALL USET ('REALNUMBER')
CALL UWRITE (58.,25.,188.)
CALL USET ('INTEGER')
CALL UPINT (75.,8., -123456789.)
CALL USET ('XYCOORDINATE')
CALL UWRITE (25.,58.,COORD)
CALL UEND
STOP
END
```

```

C   SAMPLE PROGRAM TO ILLUSTRATE OPTIONS AVAILABLE THROUGH
C   MARGINING. DEFAULT VALUES W'LL BE USED IN THIS EXAMPLE WITH
C   ADDITIONAL CALLS TO UMARGN TO ADJUST THE ALPHANUMERIC WINDOW.
C   THIS PROGRAM WAS WRITTEN FOR A TEKTRONIX 4010/4013 TERMINAL.
C
C   SET UP 300 CHARACTER ARRAY NAMED SAMPLE; PUT TEXT IN IT. NOTE
C   SEMICOLON (;)
C
C   CHARACTER SAMPLE*300
C
C   DATA SAMPLE/'THIS IS A LINE OF OUTPUT TEXT WHICH IS LONG ENOUGH TO
C   CAUSE THE ALPHANUMERIC OUTPUT TO WRAP-AROUND. NOTE THE
C   EFFECTS WHICH THE DEFAULT MARGINS HAVE UPON OUTPUT;'/
C   INITIALIZE
C
C   CALL USTART
C
C   NOW PRINT IT WITH FIRST CHARACTER OF PRINT STRING AT COORDINATE
C   LOCATION (20,25)
C
C   CALL USET ('DEVICE')
C   CALL USET ('PERCENTUNITS')
C   CALL UPRINT (20.,25.,SAMPLE)
C
C   SECOND EXAMPLE
C
C   EXAMPLE ILLUSTRATING THE USEFULNESS OF MARGINING. THE MARGINS
C   WILL BE SET USING 'FONTUNIT' COORDINATE ADDRESSING WITH THE LEFT
C   MARGIN AT APPROXIMATELY MID-SCREEN, I.E. 35 CHARACTER-WIDTHS OR
C   FONTUNITS FROM THE LEFT EDGE, AND THE RIGHT MARGIN 1 FONTUNIT
C   LATER AT POSITION 36, GIVING A TOTAL MARGIN WIDTH OF ONE
C   CHARACTER.
C
C   CALL UPSET ('TERMINATOR', ';')
C   CALL USET ('FONTUNIT')
C   CALL UMARGN (35.,36.,1.,35.)
C
C   POSITION PRINTING AT TOP OF PAPER. IN THIS EXAMPLE WE SPECIFY 4000
C   CHARACTER-HEIGHTS (FONTUNITS) FROM BOTTOM OF PAPER, AN
C   UNREASONABLY LARGE VALUE, SO GCS DEFAULTS TO THE LARGEST
C   PHYSICALLY POSSIBLE VALUE, THE TOP OF THE PAGE.
C
C   CALL UPRINT (0.,4000.,'HELLO THERE!;')
C
C   WRAP UP
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE IV-2



HELLO THERE

THIS IS A LINE OF OUTPUT TEXT WHICH IS LONG ENOUGH TO CAUSE THE ALPHANUMERIC OUTPUT TO WRAP-AROUND. NOTE THE EFFECTS WHICH THE DEFAULT MARGINS HAVE UPON OUTPUT

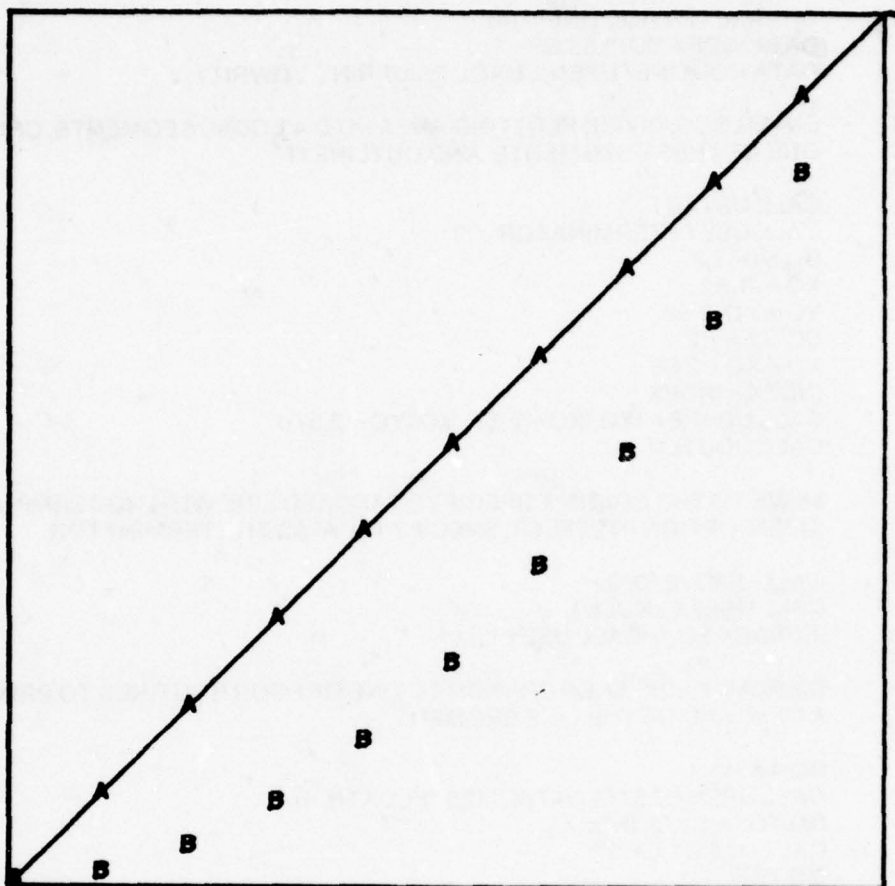
```
CHARACTER SAMPLES
DATA SAMPLE/'THIS IS A LINE OF OUTPUT TEXT
& WHICH IS LONG ENOUGH TO CAUSE THE ALPHANUMERIC
& OUTPUT TO WRAP-AROUND. NOTE THE EFFECTS WHICH THE
& DEFAULT MARGINS HAVE UPON OUTPUT,')
CALL USTART
CALL USET ('TERMINATOR',',,')
CALL USET ('PERCENTUNITS')
CALL USET ('DEVICE')
CALL UPRINT (28.,25.,SAMPLE)
CALL USET ('FONTUNITS')
CALL UMARGIN (35.,35.,1.,35.)
CALL UPRINT (8.,4888.,'HELLO THERE!,')
CALL UEND
STOP
END
```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE SINGLE CHARACTER OUTPUT
C   AND LINE TERMINATOR OPTIONS AVAILABLE THROUGH 'UPEN'. TWO
C   GRAPHS WILL BE PLOTTED:
C   (1) A LINEAR EQUATION ( $Y = X$ ). THE DATA FOR PLOTTING IT WILL BE
C       PRESTORED WITH THE VALUES OF THE INDEPENDENT VARIABLE X IN
C       THE ARRAY X AND CORRESPONDING VALUES OF Y IN THE ARRAY Y.
C
C   (2) A QUADRATIC EQUATION ( $Z = (.1 * X) ** 2$ ). THE DATA VALUES FOR THE
C       SAME VALUES OF THE INDEPENDENT VARIABLE X WILL BE STORED
C       IN THE Z ARRAY.
C
C   DEFAULT VIRTUAL WINDOW AND DEVICE AREA WILL BE USED. THE LINEAR
C   EQUATION WILL HAVE AN 'A' TERMINATOR AT THE END OF VISIBLE LINE
C   SEGMENT, WHEREAS THE QUADRATIC EQUATION WILL HAVE A 'B'
C   TERMINATOR FOR EACH INVISIBLE LINE SEGMENT.
C
C   FIRST SET UP X,Y, & Z ARRAYS. PRESTORE DATA IN THEM
C
C   DIMENSION X(11),Y(11),Z(11)
C   DATA X/O.,10.,20.,30.,40.,50.,60.,70.,80.,90.,100./
C   DATA Y/O.,10.,20.,30.,40.,50.,60.,70.,80.,90.,100./
C   DATA Z/O.,1.,4.,9.,16.,25.,36.,49.,64.,81.,100./
C
C   ENTER GCS AND DRAW OUTLINE OF DEFAULT WINDOW
C
C   CALL USTART
C   CALL UOUTLN
C
C   USTART HAS IMPLICITLY SET PEN TO SOLID-LINE MODE. NOW SPECIFY
C   THAT AN 'A' TERMINATOR SHOULD GO ON EVERY LINE AND MOVE TO INITIAL
C   (X,Y) POINT TO BE PLOTTED
C
C   CALL USET ('LA')
C   CALL UMOVE (X(1),Z(1))
C
C   DRAW 11 LINE SEGMENTS X(1),Y(1) TO X(2),Y(2) TO X(3),Y(3) AND SO ON TO
C   X(11),Y(11) THEREBY PLOTTING TOTAL EQUATION
C
C   DO 1 I=1,11
1  CALL UPEN (X(I),Y(I))
C
C   SET LINE TYPE TO 'NOLINE' AND SPECIFY THAT A 'B' TERMINATOR IS TO BE
C   USED FOR EACH INVISIBLE LINE SEGMENT
C
C   CALL USET ('NB')
C
C   MOVE TO FIRST (X,Z) POINT, THEN PLOT 11 (X,Z) VALUES
C
C   CALL UMOVE (X(1),Z(1))
C   DO 2 I=1,11
2  CALL UPEN (X(I),Z(I))
C
C   WRAP UP
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE IV-3



```

DIMENSION XC(11), YC(11), ZC(11)
DATA X/8., 18., 28., 38., 48., 58., 68., 78., 88., 95., 100./
DATA Y/8., 18., 28., 38., 48., 58., 68., 78., 88., 95., 100./
DATA Z/8., 1., 4., 9., 16., 25., 36., 49., 64., 81., 100./
CALL USTART
CALL UOUTLN
CALL USET ('LA')
CALL UMOVE (XC(1), ZC(1))
DO 1 I = 1, 11
1 CALL UPEN (XC(I), YC(I))
CALL USET ('NB')
CALL UMOVE (XC(1), ZC(1))
DO 2 I = 1, 11
2 CALL UPEN (XC(I), ZC(I))
CALL UEND
STOP
END

```

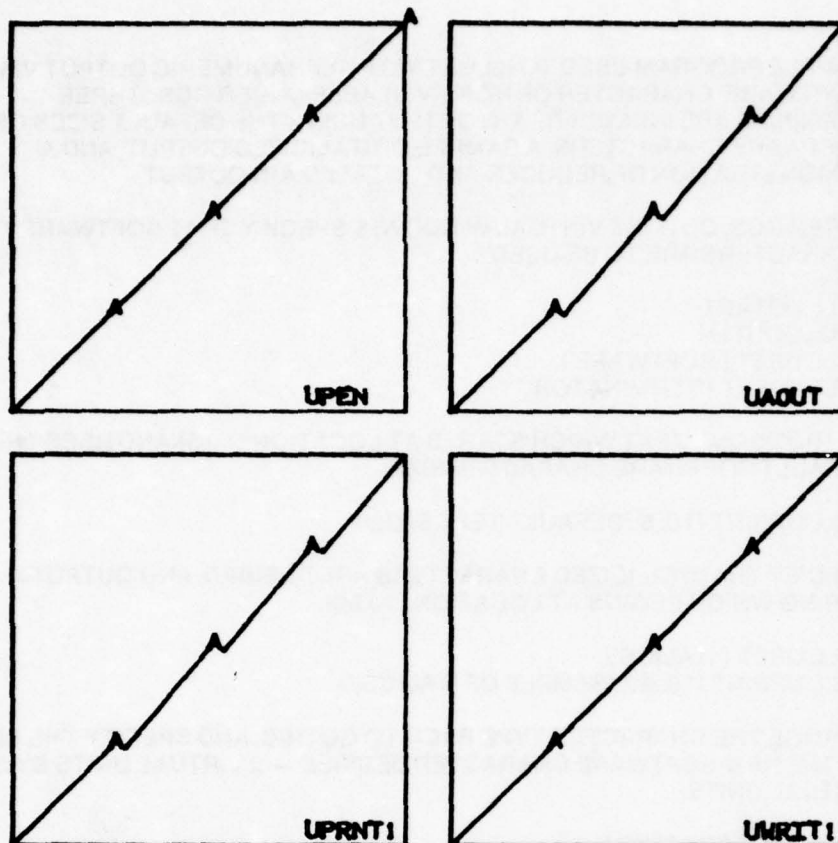


```

C   SAMPLE PROGRAM USED TO ILLUSTRATE CHARACTER TERMINATOR AND
C   A/N OUTPUT USED IN CONJUNCTION WITH GRAPHICAL OUTPUT FOR A
C   TEKTRONIX 4010/4013 TERMINAL.
C
C   CHARACTER ROUTINE*7(4)
C   DATA INDEX,YO/0,5.73/
C   DATA ROUTINE/'UPEN','UAOUT','UPRINT','UWRIT1;'/
C
C   ENTER GCS, DIVIDE PLOTTING AREA INTO 4 EQUAL SEGMENTS, CHOOSE
C   ONE OF THESE SEGMENTS, AND OUTLINE IT.
C
C   CALL USTART
C   CALL USET ('TERMINATOR',';')
C   DO 5 I=1,2
C   XO=-1.82
C   YO=YO-2.86
C   DO 5 J=1,2
C   XO=XO+2.86
C   INDEX=INDEX+1
C   CALL UDAREA (XO,(XO+2.57),YO,(YO+2.57))
C   CALL UOUTLN
C
C   MOVE TO THE ORIGIN & SPECIFY STANDARD LINE WITH NO TERMINATOR. IF
C   'UPEN' OPTION IN EFFECT, SPECIFY AN 'A' AS THE TERMINATOR.
C
C   CALL UMOVE (0.,0.)
C   CALL USET ('LNULL')
C   IF(INDEX.EQ.1) CALL USET ('LA')
C
C   DISPLAY A LINE THEN BRANCH TO ONE OF FOUR ROUTINES TO PRINT AN 'A'
C   AT THE END OF THE LINE SEGMENT.
C
C   DO 4 K=1,4
C   CALL UPEN ((25.*FLOAT(K)),(25.*FLOAT(K)))
C   GO TO (4,1,2,3), INDEX
C   1 CALL UAOUT ('A;')
C   GO TO 4
C   2 CALL UPRNT1 ('A;', 'TEXT')
C   GO TO 4
C   3 CALL UWRIT1 ('A;', 'TEXT')
C   4 CONTINUE
C
C   DISPLAY THE NAME OF THE ROUTINE WHICH WAS USED AT BOTTOM RIGHT
C   CORNER OF WINDOW.
C
C   CALL UPRINT (75.,2.,ROUTINE(INDEX))
C   5 CONTINUE
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THIS FORTRAN PROGRAM
C
C   CALL UEND
C   STOP
C   END

```

EXAMPLE IV-4



```

CHARACTER ROUTINE=7(4)
DATA INDEX,Y8/8,5.73/
DATA ROUTINE/'UPEN','UAOUT','UPRNT1','UWRIT1',/
CALL USTART
CALL UPSET ('TERMINATOR',',')
DO 5 I = 1, 2
  XB = -1.82
  Y8 = Y8 - 2.86
  DO 5 J = 1, 2
    XB = XB + 2.86
    INDEX = INDEX + 1
    CALL UDAREA (XB,(XB+2.57),Y8,(Y8+2.57))
    CALL UOUTLN
    CALL UMOVE (8.,8.)
    CALL USET ('LNULL')
    IF (INDEX.EQ. 1) CALL USET ('LA')
    DO 4 K = 1, 4
      CALL UPEN ((25.*FLOAT(K)),(25.*FLOAT(K)))
      GO TO (4,1,2,3), INDEX
1    CALL UAOUT ('A,')
      GO TO 4
2    CALL UPRNT1 ('A','TEXT')
      GO TO 4
3    CALL UWRIT1 ('A','TEXT')
4    CONTINUE
      CALL UPRNT1 (75.,2.,ROUTINE(INDEX))
5    CONTINUE
      CALL UEND
      STOP
      END

```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE ALPHANUMERIC OUTPUT VIA THE
C   'SOFTWARE' CHARACTER OPTION AVAILABLE UNDER GCS. THREE
C   EXAMPLES ARE INCLUDED: A/N OUTPUT USING THE DEFAULT SIZES OF
C   SOFTWARE CHARACTERS, A SAMPLE OF ITALICIZED OUTPUT, AND A
C   DEMONSTRATION OF REDUCED AND ROTATED A/N OUTPUT.
C
C   ENTER GCS, OUTLINE VIRTUAL WINDOW, & SPECIFY THAT SOFTWARE
C   CHARACTERS ARE TO BE USED
C
C   CALL USTART
C   CALL UOUTLN
C   CALL USET ('SOFTWARE')
C   CALL UPSET ('TERMINATOR', ';')
C
C   OUTPUT SOME TEXT WHICH STARTS AT LOCATION (10,5) AND USES THE
C   DEFAULT SOFTWARE CHARACTER SIZE.
C
C   CALL UPRINT (10.,5., 'DEFAULT TEXT SIZE;')
C
C   SPECIFY THAT ITALICIZED CHARACTERS ARE DESIRED, AND OUTPUT A
C   STRING WHICH BEGINS AT LOCATION (10,90).
C
C   CALL USET ('ITALICS')
C   CALL UPRINT (10.,90., 'SAMPLE OF ITALICS;')
C
C   CHANGE THE CHARACTER TYPE BACK TO GOTHIC, AND SPECIFY THE SIZE
C   OF THE NEW SOFTWARE CHARACTER DESIRED — 2 VIRTUAL UNITS BY 3
C   VIRTUAL UNITS.
C
C   CALL USET ('GOTHIC')
C   CALL UPSET ('HORIZONTAL', 2.)
C   CALL UPSET ('VERTICAL', 3.)
C
C   MOVE TO THE ORIGIN OF THE CURRENT COORDINATE SYSTEM, AND
C   PERFORM A 45 DEGREE ROTATION ABOUT THIS ORIGIN.
C
C   CALL UMOVE (0.,0.)
C   CALL UROTAT (45.)
C
C   OUTPUT A STRING WHICH BEGINS AT LOCATION (40,0) WITHIN THE
C   ROTATED COORDINATE SYSTEM.
C
C   CALL UPRINT (40.,0., 'REDUCED AND ROTATED CHARACTERS;')
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THE FORTRAN PROGRAM.
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE IV-5



*SAMPLE OF ITALICS*

REDUCED AND ROTATED CHARACTERS

DEFAULT TEXT SIZE

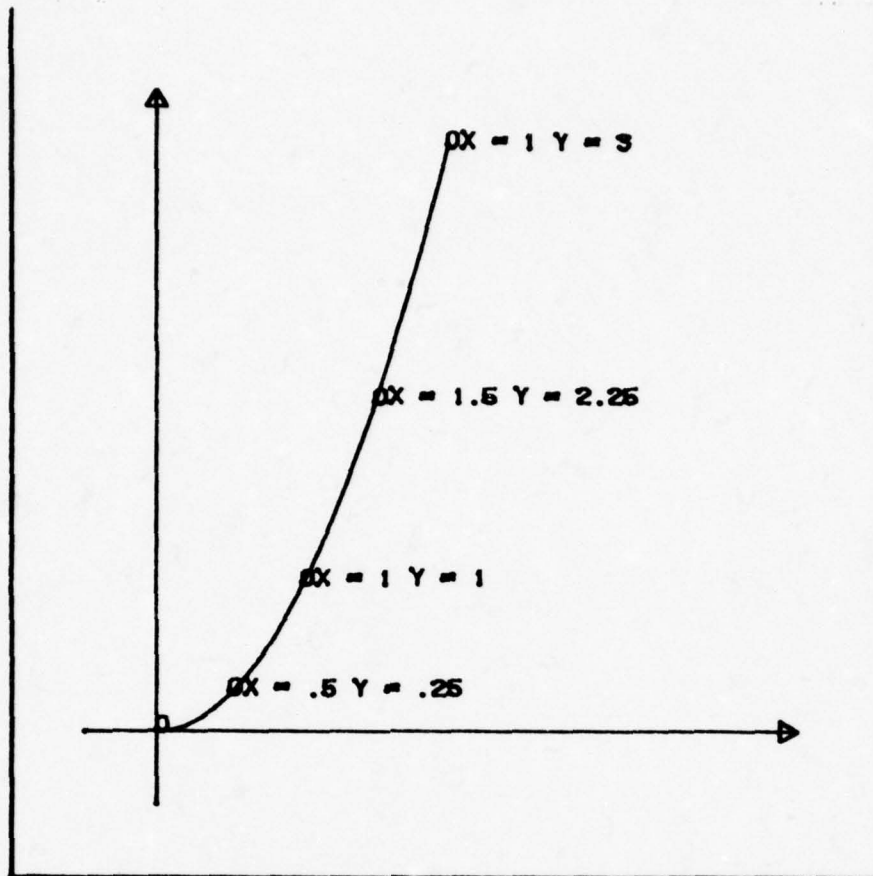
```
CALL USTART
CALL UOUTLN
CALL UPSET ('TERMINATOR',',',')
CALL USET ('SOFTWARE')
CALL UPRINT (18.,5., 'DEFAULT TEXT SIZE,')
CALL USET ('ITALICS')
CALL UPRINT (18.,28., 'SAMPLE OF ITALICS,')
CALL USET ('GOTHIC')
CALL UPSET ('HORIZONTAL',2.)
CALL UPSET ('VERTICAL',3.)
CALL UMOVE (8.,8.)
CALL UROTAT (45.)
CALL UPRINT (48.,8., 'REDUCED AND ROTATED CHARACTERS,')
CALL UEND
STOP
END
```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE 'UPRINT' TO OUTPUT MIXED
C   ALPHANUMERIC DATA
C
C   INITIALIZE ALL VARIABLES WHICH WILL BE USED
C
C   CHARACTER OPTION*10(4)/'REALNUMBER','INTEGER','REALNUMBER',
C   'INTEGER'/
C   CHARACTER TITLE*6(2)/'X=','Y='//
C   DATA X,Y,/0.,0./
C
C   ENTER GCS SET UP WINDOW AND DRAW AXES
C
C   CALL USTART
C   CALL UOUTLN
C   CALL UWINDO (-1.,5.,-1.,5.)
C   CALL UMOVE (0.,-.5)
C   CALL UPEN1 (0.,4.4.,'L'ARROW')
C   CALL UMOVE (-.5,0)
C   CALL UPEN (4.4,0.,'L'ARROW')
C
C   MOVE TO ORIGIN AND MARK IT
C
C   CALL UPEN (X,Y,'NO')
C
C   DRAW A SEGMENT OF A PARABOLA, MARK THE COORDINATES AT
C   INTERVALS OF .5 FOR X AND ALTERNATELY OUTPUT THE VALUES OF X AND
C   Y AS REAL AND INTEGER.
C
C   DO 1 I=1,4
C   DO 2 J=1,5
C   X=X+ .10
C   Y=X**2
C   CALL UPEN (X,Y)
C   2 CONTINUE
C   CALL UPEN1 (X,Y,'NO')
C   CALL UPRNT1 (TITLE(1),'TEXT')
C   CALL UPRNT1 (X,OPTION(1))
C   CALL UPRNT1 (TITLE(2),'TEXT')
C   CALL UPRNT1 (Y,OPTION(1))
C   CALL UMOVE (X,Y)
C   1 CONTINUE
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE PROGRAM.
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE IV-6



```

CHARACTER OPTION=18(4)
DATA OPTION/'REALNUMBER','INTEGER',
&          'REALNUMBER','INTEGER'/
CHARACTER TITLE=6(2)/' X = ',' Y = ',/
DATA X,Y/0.,0./
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UOUTLN
CALL UINDO (-1.,5.,-1.,5.)
CALL UMOVE (0.,-.5)
CALL UPEN1 (0.,4.4,'LARROW')
CALL UMOVE (-.5,0.)
CALL UPEN1 (4.4,0.,'LARROW')
CALL UPEN1 (X,Y,'NO')
DO 1 I = 1, 4
DO 2 J = 1, 5
X = X + .10
Y = X**2
CALL UPEN (X,Y)
2 CONTINUE
CALL UPEN1 (X,Y,'NO')
CALL UPRT1 (TITLE(1),'TEXT')
CALL UPRT1 (X,OPTION(1))
CALL UPRT1 (TITLE(2),'TEXT')
CALL UPRT1 (Y,OPTION(2))
CALL UMOVE (X,Y)
1 CONTINUE
CALL UEND
STOP
END

```



## **CHAPTER V**

### **GRAPHICAL AND ALPHANUMERIC INPUT**

The following section discusses the facilities within GCS that are available for entering data of a graphical nature. Not all implementations of GCS can be used for such graphics input. For example, those implementations of GCS running in a batch mode, i.e., **passive graphics**, do not permit graphical input. For these applications, the routines discussed below are disabled.

For a graphics system to be truly interactive, it is necessary to provide the user with the ability of communicating information of a strictly graphical nature to his graphics program. Graphical input differs from conventional FORTRAN input (i.e., READ, etc.) in the following areas:

- A. Graphical input is normally performed as a composite operation involving the transfer of positional coordinate information from the terminal's primary graphic input device (cursor, light pen, etc.) in conjunction with a single character from the keyboard (or other suitable interrupt) which signals the termination of the input operation.
- B. Because of the unique nature of strictly graphical input, standard FORTRAN input cannot be used in lieu of software which has been specifically designed for that given purpose.
- C. Graphical input does not produce any extraneous output at the graphics terminal when entering information, in contrast with standard FORTRAN free-formatted input which alerts the user through the use of a special symbol that input is requested, and echoes the input information as it is entered.

#### **Positional and Character Graphical Input**

The standard positional and character graphics input subroutine implemented within GCS is UGRIN, which may be called by the following sequence:

**CALL UGRIN (X,Y,CHAR)**

When UGRIN is invoked, the terminal's primary graphics input device is activated, and remains in an enabled state until a character is entered from the keyboard. Upon receipt of the character, UGRIN stores it into CHAR using standard FORTRAN Hollerith format, disables the primary input device, and transfers the positional coordinates (in current units) of the input device at the time the character was entered into X and Y. See Example V-1.

Subroutine UAIN functions in the same manner as UGRIN, except that no positional information is returned. Since the terminal remains in 'limbo' while awaiting input from the user, UAIN provides a convenient method of pausing at desired points during execution of a graphics program.

**CALL UAIN (CHAR)**

#### **Alphanumeric Input (Fortran)**

For bulk alphanumeric input used in conjunction with interactive communications and

coordination activities, standard FORTRAN may be utilized under GCS using the same criteria outlined in Chapter IV for FORTRAN alphanumeric output; i.e., the terminal must be placed into the alphanumeric mode through the call to UALPHA before control is relinquished to the FORTRAN input routines. It should be emphasized, however, that although FORTRAN input is extremely convenient, the use of FORTRAN input procedures significantly reduces the efficiency at which a graphics program executes; hence, the use of FORTRAN I/O should be avoided whenever possible.

#### **Alphanumeric Input (Graphical)**

An alternative method of alphanumeric input is provided to the graphics programmer through subroutine UREAD:

#### **CALL UREAD (X,Y,DATA,COUNT,FLAG)**

X and Y are used to specify the starting position (in current units) of where the alphanumeric input is to be attempted. UREAD will store the edited input into DATA, based upon one of the four options available to the user through a call of USET prior to invoking UREAD.

- A. **TEXT** — Under this default option, UREAD will accept and store COUNT characters into DATA; hence, the user must insure that DATA has been suitably dimensioned to hold the number of characters which have been requested. Should fewer than COUNT characters be entered as input, UREAD will store the actual number of characters entered into FLAG, and blank-fill the remaining (COUNT - FLAG) characters of DATA. It should be noted that UREAD does not append the termination character to the end of the input. Therefore, the user is responsible for inserting this character via UAPEND if the string is to be passed as a parameter to UPRINT or UWRITE.
- B. **REALNUMBER** — This option directs UREAD to edit the alphanumeric input as a REAL number, and to store the resulting floating-point number into the single-valued REAL parameter DATA. Should UREAD encounter any illegal characters during the edit, FLAG will be returned with a negative value, and DATA will be undefined.
- C. **INTEGER** — As in the case of REALNUMBER, DATA is assumed to be single-valued and of type REAL. UREAD will edit the alphanumeric input as an INTEGER, perform an INTEGER to REAL conversion, and store the result into DATA. The user may check if the operation was successfully performed by examining FLAG upon return from UREAD.
- D. **XYCOORDINATES** — This option directs UREAD to accept two REAL numbers (separated by a comma) as input and to store them into the two element REAL array, DATA. FLAG is set to reflect the status of the input and editing operation.

Under **REALNUMBER**, **INTEGER**, and **XYCOORDINATES** options, COUNT is used to specify the number of variables which are to be input under that given mode. For example, if two integers were desired to be read into DATA at virtual location (0.100.), the following call to UREAD could be used:

```
CALL USET ('INTEGER')  
CALL UREAD (0.,100.,DATA,2.,FLAG)
```

Under **XYCOORDINATES**, COUNT designates the number of ordered pairs to be input.

A streamlined version of UREAD is available to the user under GCS through subroutine UINPUT:

### **CALL UINPUT (DATA,COUNT,FLAG,OPTION)**

DATA, COUNT and FLAG are interpreted in the same manner as UREAD; OPTION is a single-valued character variable which specifies the format or mode under which the input is to be edited and stored into DATA, i.e., TEXT, REALNUMBER, INTEGER, etc. The UINPUT/UREAD relationship closely parallels UPRNT1/UPRINT in that the operation is performed at the current beam position, and the effects of OPTION are purged from the GCS upon exit from the subroutine. See Example V-2.

### **Menued Graphical Input**

Aside from entering positional information which directs the generation of a display, the bulk of most interactive input requirements centers upon the ability to select one of several options which are 'menued' before the user. Menuing represents a very attractive means of user interaction due to the inherent simplicity of the required user response -- one need only position the graphic input device within the square of the menuboard which represents the desired option, and depress a character on the keyboard to transmit the selection to the user's program.

Within GCS, graphical input, by means of a menuboard, is facilitated through the use of subroutine UMENU, which is called by the following sequence:

### **CALL UMENU (PTSIN,LABELS,CHOICE)**

The absolute value of PTSIN represents the number of menu options that will be provided (maximum of 10); LABELS is a character array containing one entry of up to eight characters which is to be printed under each menu option; and CHOICE is a single-valued output REAL variable, assigned a value by UMENU which indicates the box number of the option which the user has selected. When UMENU is called with a positive value of PTSIN, a menuboard is drawn, and the graphic input device is enabled. A negative value of PTSIN merely enables the input device and inhibits the redrawing of the menuboard. See Example V-3.

### **Drafting-Type Graphical Input**

The three GCS graphical input subroutines described in this chapter represent a core around which many sophisticated interactive graphical programs may be designed. There exists one additional GCS graphical input subroutine, UDRIN, which permits high-level drafting-type activities to be performed under program control; in addition, the result of the man/machine interaction may be saved and recalled during future interactions.

### **CALL UDRIN(X,Y,CHAR)**

Interested users are directed to the various subroutine descriptions for further discussion.

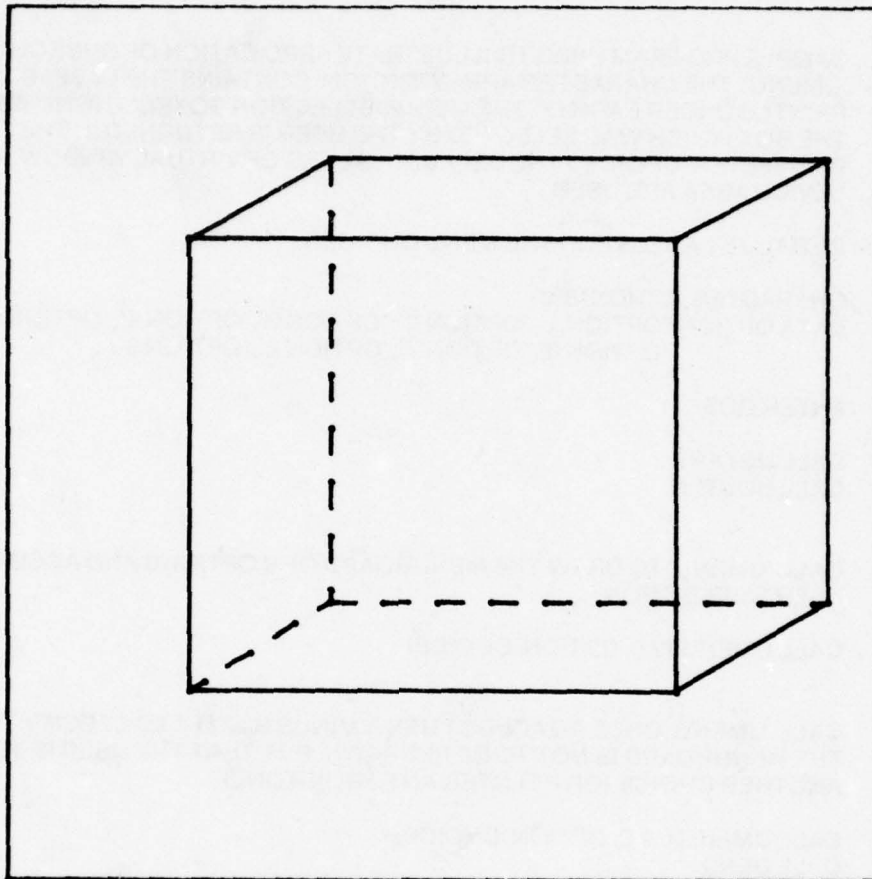


```

C
C      SAMPLE PROGRAM WHICH ILLUSTRATES GRAPHICS INPUT APPLICATION
C      THROUGH 'UGRIN'. THREE TYPES OF CASES ARE HANDLED: SOLID 'S'
C      INVISIBLE 'I', AND DASHED 'D' LINES. THE DESIRED OPTION FOR THE LINE IS
C      ENTERED AS A SINGLE CHARACTER WHEN THE CURSORS HAVE BEEN
C      POSITIONED. DEFAULT VALUES OF WINDOW, DEVICE AREA, AND DASH
C      SPECIFICATION ARE USED. AN 'E' WILL TERMINATE THE PROGRAM.
C
C      CHARACTER CHAR*1
C      CALL USTART
C      CALL UOUTLN
C
C      ENABLES CURSORS, OBTAIN (X,Y) COORDINATES, AND THE CHARACTER.
C
C      1  CALL UGRIN (X,Y,CHAR)
C
C      CHECK FOR A REQUEST FOR A SOLID LINE TO BE DRAWN TO (X,Y).
C
C      IF (CHAR.EQ.'S') CALL UPEN1 (X,Y,'LINE')
C
C      CHECK FOR A REQUEST FOR AN INVISIBLE LINE (MOVE) TO (X,Y).
C
C      IF (CHAR.EQ.'I') CALL UMOVE (X,Y)
C
C      CHECK FOR A REQUEST TO DRAW A DASHED LINE TO POINT (X,Y).
C
C      IF (CHAR.EQ.'D') CALL UPEN1 (X,Y,'DASH')
C
C      CHECK IF THE USER DESIRES TO TERMINATE CURRENT GCS PROGRAM.
C
C      IF (CHAR.EQ.'E') GO TO 2
C      GO TO 1
C      2  CALL UEND
C      STOP
C      END

```

#### EXAMPLE V-1



```

CHARACTER CHAR=1
CALL USTART
CALL UOUTLN
1 CALL UORDIN CX,Y,CHAR)
  IF (CHAR .EQ. 'S') CALL UPEN1 CX,Y,'LINE')
  IF (CHAR .EQ. 'I') CALL UMOVE CX,Y)
  IF (CHAR .EQ. 'D') CALL UPEN1 CX,Y,'DASH')
  IF (CHAR .EQ. 'E') GO TO 2
  GO TO 1
2 CALL UEND
STOP
END

```

```

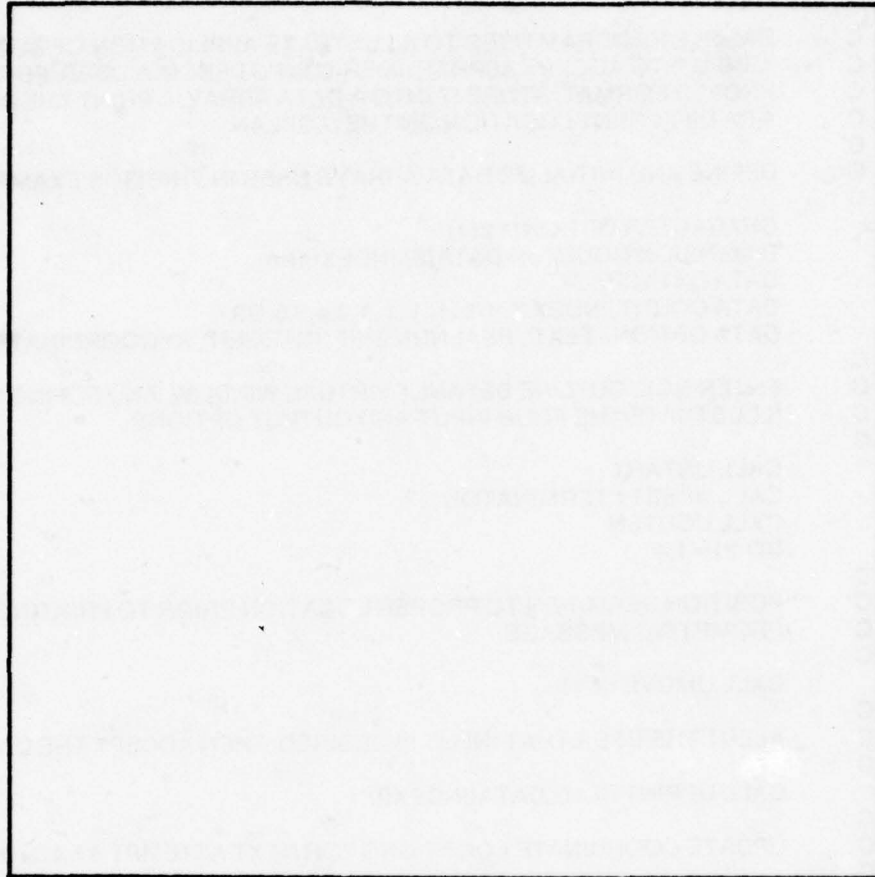
C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF SUBROUTINE
C   'UMENU'. THE CHARACTER ARRAY 'OPTION' CONTAINS THE LABELS TO BE
C   PRINTED UNDER EACH OF THE MENUE SELECTION BOXES. THE NUMBER OF
C   THE BOX WHICH WAS SELECTED BY THE USER IS RETURNED IN THE
C   PARAMETER 'CHOICE'. THE DEFAULT VALUES OF VIRTUAL WINDOW AND
C   DEVICE AREA ARE USED.
C
C   INITIALIZE LABELS FOR THE MENU CHOICES
C
C   CHARACTER OPTION*8(9)
C   DATA OPTION/'OPTION 1','OPTION 2','OPTION 3','OPTION 4','OPTION 5',
C   'OPTION 6','OPTION 7','OPTION 8','OPTION 9'/
C
C   ENTER GCS
C
C   CALL USTART
C   CALL UOUTLN
C
C   CALL 'UMENU' TO DRAW THE MENUBOARD OF 9 OPTIONS AND ACCEPT THE
C   USER'S SELECTION
C
C   CALL UMENU (9.0, OPTION, CHOICE)
C
C   CALL 'UMENU' ONCE AGAIN, BUT USE A MINUS SIGN (-) TO SPECIFY THAT
C   THE MENUBOARD IS NOT TO BE REDRAWN, BUT THAT THE USER IS TO INPUT
C   ANOTHER CHOICE (OR REENTER AN EARLIER ONE)
C
C   CALL UMENU (-9.0, OPTION, CHOICE)
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE V-2



☐ 8  
 OPTION 8  
☐ 8  
 OPTION 8  
☐ 7  
 OPTION 7  
☐ 6  
 OPTION 6  
☐ 5  
 OPTION 5  
☐ 4  
 OPTION 4  
☐ 3  
 OPTION 3  
☐ 2  
 OPTION 2  
☐ 1  
 OPTION 1



```

CHARACTER OPTION=8(8)
DATA OPTION/'OPTION 1','OPTION 2','OPTION 3','OPTION 4',
&          'OPTION 5','OPTION 6','OPTION 7','OPTION 8',
&          'OPTION 9'/
CALL USTART
CALL UCUTLN
CALL UMENU (8.8,OPTION,CHOICE)
CALL UMENU (-8.8,OPTION,CHOICE)
CALL UEND
STOP
END
  
```

```

C      SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF SUBROUTINE
C      'UINPUT' TO ACCEPT ALPHANUMERIC INPUT FROM A USER, EDIT IT INTO THE
C      PROPER FORMAT, STORE IT INTO A DATA ARRAY, & PRINT THE DATA ARRAY
C      AT A DIFFERENT LOCATION ON THE DISPLAN.
C
C      DEFINE AND INITIALIZE DATA ARRAYS USED IN THIS GCS EXAMPLE
C
C      CHARACTER OPTION*12(4)
C      DIMENSION COUNT(4),DATA(6),INDEX(4)m
C      DATA DATA(2)"/'/'/
C      DATA COUNT, INDEX,X,Y/5.,1.,1.,1.,3,4,5,5.,90./
C      DATA OPTION/"TEXT",'REALNUMBER','INTEGER','XYCOORDINATE'/
C
C      ENTER GCS, OUTLINE DEFAULT VIRTUAL WINDOW, AND DEFINE LOOP TO
C      ILLUSTRATE THE FOUR INPUT AND OUTPUT OPTIONS.
C
C      CALL USTART
C      CALL UPSET ('TERMINATOR',';')
C      CALL UOUTLN
C      DO 1 I=1,4
C
C      POSITION BEAM/PEN TO PROPER LOCATION PRIOR TO PRINTING OF A
C      PROMPTING MESSAGE.
C
C      CALL UMOVE (X,Y)
C
C      ALERT THE USER THAT INPUT IS DESIRED, THEN ACCEPT THE DATA
C
C      CALL UPRINT (X,10.,DATA(INDEX(I)))
C
C      UPDATE COORDINATE LOCATIONS FOR NEXT ATTEMPT AT A/N INPUT
C
C      X=X+22.5
C      Y=Y-20.
C      1 CONTINUE
C      WRAP-UP GRAPHICS ACTIVITY, & TERMINATE THE FORTRAN PROGRAM.
C
C      CALL UEND
C      STOP
C      END

```

#### EXAMPLE V-3

ENTER: GREETINGS

ENTER: 1.234E+10

ENTER: -123456789

ENTER: 1.2,3.4

GREET            .1234E+11    -123456789    (1.2,3.4)

```
CHARACTER OPTION=12(4)
DIMENSION COUNT(4),DATA(6),INDEX(4)
DATA COUNT,INDEX,X,Y/5.,1.,1.,1.,1,3,4,5,5.,88./
DATA OPTION/'TEXT','REALNUMBER','INTEGER','XYCOORDINATE'/
CALL USTART
CALL USET ('TERMINATOR',',')
CALL UOUTLN
DO I = 1, 4
CALL UMOVE (X,Y)
CALL UPRINT ('ENTER: ',',','TEXT')
CALL UINPUT (DATAINDEX(I)),COUNT(I),FLAG,OPTION(I))
IF(I.EQ.1) CALL UAPEND(COUNT(I),DATAINDEX(I)),DATAINDEX(I))
CALL USET (OPTION(I))
CALL UPRINT (X,10.,DATAINDEX(I))
X = X + 22.5
Y = Y - 28.
1 CONTINUE
CALL UEND
STOP
END
```



## CHAPTER VI

### GCS UTILITY SUBROUTINES

To assist the user in preparing his graphical applications, GCS has incorporated a series of comprehensive, yet easy to use utility subroutines which permit the programmer to generate:

- A. Circles and circular arcs
- B. Regular polygons and other straight-sided geometric figures
- C. Conic sections
- D. Linear and least-squares polynomial curve fits to a series of user-supplied data points.

All of the GCS utility routines which generate graphical output perform their activities within the environment established by the user at the time they were invoked; hence, such output will be constrained by such parameters as the virtual window and device area, current units and mode of addressing, line type and terminator, together with any rotational and scaling factors specified by the user.

#### Circles

One of the most common requirements of scientific and engineering graphical applications is for the generation of circles and circular arcs. Provision for this capability under GCS is available through subroutines UCRCL and UARC.

Circles may be generated through subroutine UCRCL which is called by the following sequence:

#### CALL UCRCL (X,Y,RADIUS)

X and Y are used to specify the coordinates (in current units) of the center of the circle of radius, RADIUS, which will be approximated by a series of line segments forming a many-sided equilateral polygon, whose peak deviation from an ideal circle is approximately 1.4 thousandths of the ideal circle's radius. This error is normally comparable in magnitude to the raster spacing of the plotting device, and should prove adequate under most conditions. Example VI-1 demonstrates the use of UCRCL.

#### Circular Arcs

Circular arcs may be generated through subroutine UARC which is called by the following sequence:

#### CALL UARC (X,Y,ANGLE)

Upon invocation, UARC will generate an arc beginning at the current beam position and of angular span ANGLE. The radius of the arc is determined through computation of the distance from the center of the arc (specified by X and Y) and the beam position prior to entry into the subroutine. Upon exit, UARC will leave the beam at the last point of the angular segment it has generated. For example, if all default conditions are in effect, the current beam position is at (40.0,50.0), and UARC is invoked with the following parameters:

### **CALL UARC (40.0,20.0,90.0)**

UARC will generate a circular arc having a center at (40.0,20.0), with a radius of 30.0, and angular span of 90.0 degrees. The arc will begin at (40.0,50.0) and terminate at (10.0,20.0). See Example VI-2.

*Should the user be at a terminal which has the capability to generate circles and circular arcs via hardware, UCRCLE and UARC will utilize such facilities when they are invoked.*

### **Regular Polygons**

Polygon and other straight-sided geometric figure generation is provided through subroutines UPLYGN and URECT. Subroutine URECT may be utilized when a reactangle, which spans the area defined by the current beam position and coordinates (X,Y), is desired. Should URECT be called under the RELATIVE mode of addressing, any rotational or scaling parameters which may apply will dictate the rotation or scaling of the rectangle about the current beam position.

### **CALL RECT(X,Y)**

Provision for the generation of equilateral (regular) polygons is available under GCS through UPLYGN which may be called by the following sequence:

### **CALL UPLYGN (X,Y,PTS,RADIUS)**

X and Y denote the coordinates of the center of the polygon, PTS is the number of sides which is to comprise the figure, and RADIUS is used to define the radius of the circle within which the polygon is inscribed. With no relative rotation specified, the polygon will be drawn with one side parallel to the bottom of the plotting area. Should relative rotation be specified, this "base" side will be rotated by the given angle of rotation. In contrast to URECT, UPLYGN's axis of rotation is defined by (X,Y) as opposed to the beam position upon entry to the subroutine. Examples VI-3 through VI-6 demonstrate the use of UPLYGN and URECT and illustrates some of the principles of rotation and line options.

### **Conic Sections**

A particularly useful subroutine which may be used to draw all (or part) of a generalized conic section having a given focus, directrix, eccentricity, and angular span is available through the following calling sequence:

### **CALL UCONIC (X,Y,P,E,THETA1,THETA2)**

(X,Y) are used to specify the coordinates of the focus of the conic section; P is the distance from the focus to the directrix; E is the eccentricity, THETA1 and THETA2 represent the initial and final angles through which the conic section is to be drawn. Parameters E and P affect the generation of the conic section in the following manner:

- |                       |   |
|-----------------------|---|
| A. $E=0$              | will draw a circular arc with a center at (X,Y) and radius $P/2$ . The arc will subtend the angular range defined by THETA1 and THETA2. |
| B. $0.LT.ABS(E).LT.1$ | will generate an ellipse.   |
| C. $ABS(E)=1$         | will specify a parabola.  |
| D. $ABS(E).GT.1$      | designates that a hyperbola is to be drawn.   |

- |    |        |  |
|----|--------|--|
| E. | E.GT.0 | indicates that the major axis of the conic section is to be oriented parallel to the X-axis. |
| F. | E.LT.0 | specifies that the major axis is to be oriented along the Y-axis.                            |
| G. | P.GT.0 | defines the position of the focus to be to the right of (or below) the directrix.            |
| H. | P.LT.0 | indicates that the focus is positioned to the left of (or above) the directrix.              |

Examples VI-7 and VI-8 demonstrate the use of UCONIC.

### Curve Fitting

For those users who desire to combine data analysis through curve fitting into their graphical applications, subroutines ULINFT and ULSTSQ should provide of particular importance. ULINFT should be used when it is desired to obtain the slope (S) and Y-intercept (YI) of the linear equation,  $Y = SX + YI$ , which represents the least-squares linear fit to a number (XN) of user-supplied data points (contained in arrays X and Y). ULINFT may be called by the following sequence:

**CALL ULINFT (X,Y,SN,S,YI)**

See Example VI-9.

Subroutine ULSTSQ should be used when a 'least-squares' polynomial curve fit is desired. ULSTSQ returns the N+1 coefficients of the polynomial of degree N which represents the 'best' in the sense of least-square fit to a series of user-supplied data points. ULSTSQ may be invoked by:

**CALL ULSTSQ (X,Y,XN,COEFF)**

where X is a user-supplied array containing XN values of the independent variable; Y is an array of XN values for the dependent variable; XN is the number of values in each of the X and Y arrays; and COEFF is an output array which contains the N+1 coefficients of the polynomial which was fitted to the data. In order to specify the degree of the polynomial which is to be fitted to the data, it is necessary to call UPSET before ULSTSQ is invoked. For the generalized case outlined above, a suitable call to UPSET would be:

**CALL UPSET ('POLYNOMIAL',FLOAT(N))**

See Example VI-10.

### Development of Applications Library

There is one additional curve fitting routine available, USPLIN. This routine does a cubic spline fit to a series of data points. The quality of fit is established by the nature of the cubic spline function. The user is referred to the subroutine writeup.

**CALL USPLIN(X,Y,XN,RX,RY,RN)**

The subroutines discussed in this chapter represent a group of software which may be viewed as a miniature applications library. It is also important to remember that each of these routines represents applications of the various GCS 'user-level' subroutines



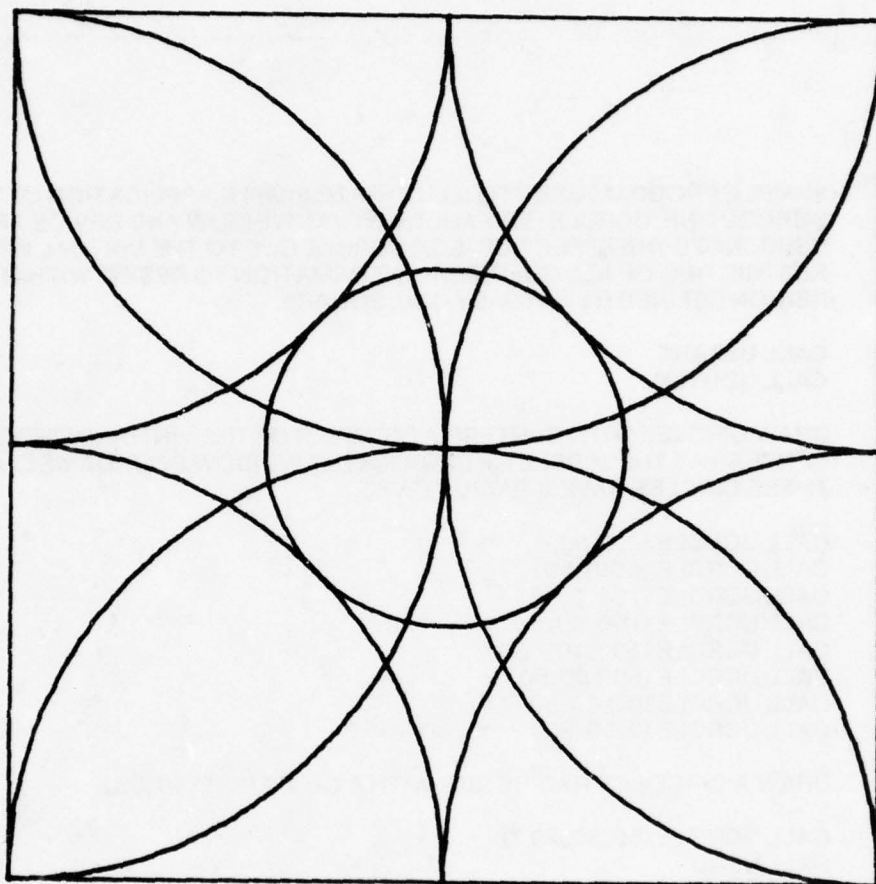
outlined in the previous chapters. In essence, subroutines such as UCRCL, URECT, and UPLYGN are representative of the types of generalized software that a sophisticated user is able to write under GCS.

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE SIMPLE APPLICATION OF THE
C   SUBROUTINE 'UCRCLE'. DEFAULT VIRTUAL WINDOW AND DEVICE AREA IS
C   USED. NOTE THE EFFECT OF 'SCISSORING' DUE TO THE VIRTUAL WINDOW'S
C   RESTRICTING OF ALL GRAPHICAL INFORMATION TO RESIDE WITHIN THE
C   REGION DEFINED BY A 100.-BY-100. SQUARE.
C
C   CALL USTART
C   CALL UOUTLN
C
C   DRAW CIRCLES WITH CENTERS AT CORNER OF THE VIRTUAL WINDOW, AND
C   CENTERS AT THE MIDDLE OF EACH OF THE WINDOW BOUNDARIES. ALL OF
C   THESE CIRCLES HAVE A RADIUS OF 50.
C
C   CALL UCRCLE (0.,0.,50.)
C   CALL UCRCLE (50.,0.,50.)
C   CALL UCRCLE (100.,0.,50.)
C   CALL UCRCLE (100.,50.,50.)
C   CALL UCRCLE (100.,100.,50.)
C   CALL UCRCLE (50.,100.,50.)
C   CALL UCRCLE (0.,100.,50.)
C   CALL UCRCLE (0.,50.,50.)
C
C   DRAW A CIRCLE OF RADIUS 20.7 WITH A CENTER AT (50.,50.).
C
C   CALL UCRCLE (50.,50.,20.7)
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VI-1



```

CALL USTART
CALL UOUTLN
CALL UCIRCLE (0.,0.,50.)
CALL UCIRCLE (50.,0.,50.)
CALL UCIRCLE (100.,0.,50.)
CALL UCIRCLE (100.,50.,50.)
CALL UCIRCLE (100.,100.,50.)
CALL UCIRCLE (50.,100.,50.)
CALL UCIRCLE (0.,100.,50.)
CALL UCIRCLE (0.,50.,50.)
CALL UCIRCLE (50.,50.,20.7)
CALL UEND
STOP
END

```

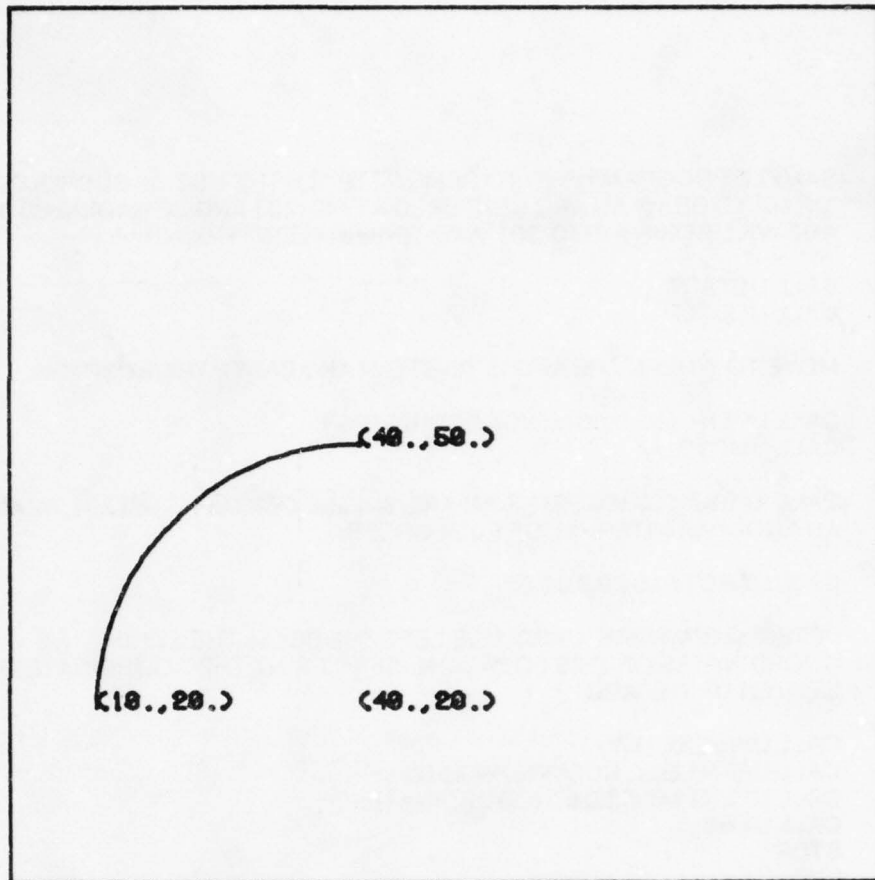


```

C      SAMPLE PROGRAM USED TO DEMONSTRATE THE USE OF SUBROUTINE
C      'URAC' TO DRAW AN ARC CENTERED AT (40.,20.) AND OF RADIUS 30.0. THE
C      ARC WILL BEGIN AT (40.,50.), AND TERMINATE AT (10.,20.).
C
C      CALL USTART
C      CALL UOUTLN
C
C      MOVE TO WHERE THE ARC IS TO BEGIN AND LABEL THE POSITION.
C
C      CALL UPEN1 (40.,0,50.0, 'NCOORDINATES')
C      CALL UMOVE
C
C      CALL 'UARC' TO GENERATE AN ARC WHOSE CENTER IS (40,20.), SPANNING
C      AN ANGULAR INTERVAL OF 90 DEGREES.
C
C      CALL UARC (40.0,20.0,90.0)
C
C      DETERMINE WHERE 'UARC' HAS LEFT THE BEAM, THEN PRINT THE
C      COORDINATES OF THIS LOCATION. ALSO PRINT THE COORDINATE OF THE
C      CENTER OF THE ARC
C
C      CALL UWHERE (X,Y)
C      CALL UPEN1 (X,Y,'NCOORDINATES')
C      CALL UPEN1 (40.0,20.0,'NCOORDINATES')
C      CALL UEND
C      STOP
C      END

```

#### EXAMPLE VI-2



```

CALL USTART
CALL UOUTLN
CALL UPEN1 (40.0,50.0,'NCOORDINATES')
CALL UMOVE (40.0,50.0)
CALL UARC (40.0,20.0,90.0)
CALL UWHERE (X,Y)
CALL UPEN1 (X,Y,'NCOORDINATES')
CALL UPEN1 (40.0,20.0,'NCOORDINATES')
CALL UEND
STOP
END

```

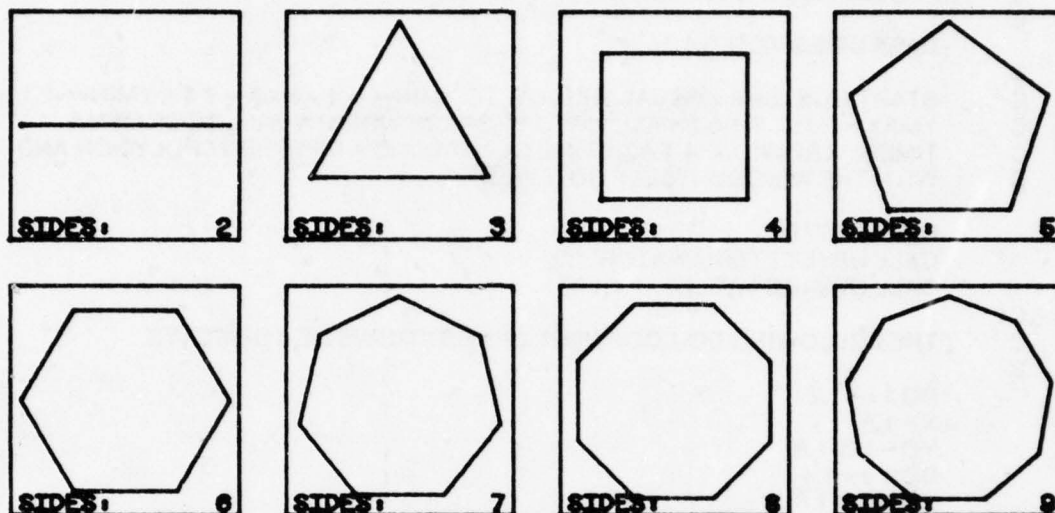
```

C   SAMPLE PROGRAM USED TO DEMONSTRATE USE OF SUBROUTINE
C   'UPLYGN'. POLYGONS OF FROM 2 TO 9 SIDES WILL BE DRAWN WITHIN THEIR
C   OWN WINDOW, WHICH IS MAPPED TO DIFFERENT PORTIONS OF THE
C   SCREEN OF A TEKTRONIX 4010/4013 TERMINAL.
C
C   PRESET NUMBER OF SIDES OF POLYGON = 1, 'INITIAL Y HEIGHT = 4.7'
C
C   DATA SIDES YO/1.0,4.7/
C
C   START GCS. SET VIRTUAL WINDOW TO XMIN=-1.1 XMAX=+1.1 YMIN=-1.1
C   YMAX=+1.1. THIS SMALL VIRTUAL-SPACE WINDOW WILL BE DRAWN 8
C   TIMES: 2 ROWS OF 4, EACH TIME CONTAINING A DIFFERENT POLYGON AND
C   WITH THE WINDOW ITSELF OUTLINED.
C
C   CALL USTART
C   CALL UPSET ('TERMINATOR', ',')
C   CALL UWINDO (-1.1,1.1,-1.1,1.1)
C
C   THE FOLLOWING DO-LOOPS SET UP THE 2 ROWS OF 4 DISPLAYS
C
C   DO 1 I=1,2
C   X=1.5
C   YO=YO-1.8
C   DO 1 J=1,4
C   XO=XO+1.8
C
C   INITIALIZE NUMBER OF SIDES, 1 IS ADDED BEFORE EACH EXECUTION SO
C   POLYGON SIDES START AT 2 AND GO TO 9 IN 8 STEPS
C
C   SIDES=SIDES+1.0
C
C   NOW ACTUALLY SET UP THE DEVICE AREA WINDOW AS XO AND YO CHANGE
C   IT WILL MOVE TO 8 DIFFERENT LOCNS
C
C   CALL UDAREA (X),(XO+1.5),YO,(YO+1.5))
C
C   AT EACH LOCATION OUTLINE IT
C
C   CALL UOUTLN
C
C   CALL 'UPLYGN' TO DRAW THE POLYGON WITHIN THE DEVICE AREA WE HAVE
C   DEFINED, THEN ADD SOME LABELING VIA CALLS TO 'UPRINT'.
C
C   CALL UPLYGN (0.0,0.0,SIDES,1.0)
C
C   LABEL DRAWING
C
C   CALL USET ('TEXT')
C   CALL UPRINT (-1.0,-1.05,'SIDES;')
C   CALL USET ('INTEGER')
C   CALL UPRINT (0.9,-1.05,SIDES)
C
1  CONTINUE
C   CALL UEND
C   STOP
C   END

```

EXAMPLE VI-3





```

DATA SIDES,Y8/1.8,4.7/
CALL USTART
CALL UPSET ('TERMINATOR','(',')')
CALL UWINDO (-1.1,1.1,-1.1,1.1)
DO I = 1, 2
  X8 = -1.5
  Y8 = Y8 - 1.8
  DO J = 1, 4
    X8 = X8 + 1.8
    SIDES = SIDES + 1.8
  CALL UDAREA (X8,X8+1.5),Y8,(Y8+1.5))
  CALL UOUTLN
  CALL UPLYGN (8.8,8.8,SIDES,1.8)
  CALL USET ('TEXT')
  CALL UPRINT (-1.8,-1.85,'SIDES:',')
  CALL USET ('INTEGER')
  CALL UPRINT (8.8,-1.85,SIDES)
1 CONTINUE
CALL UEND
STOP
END

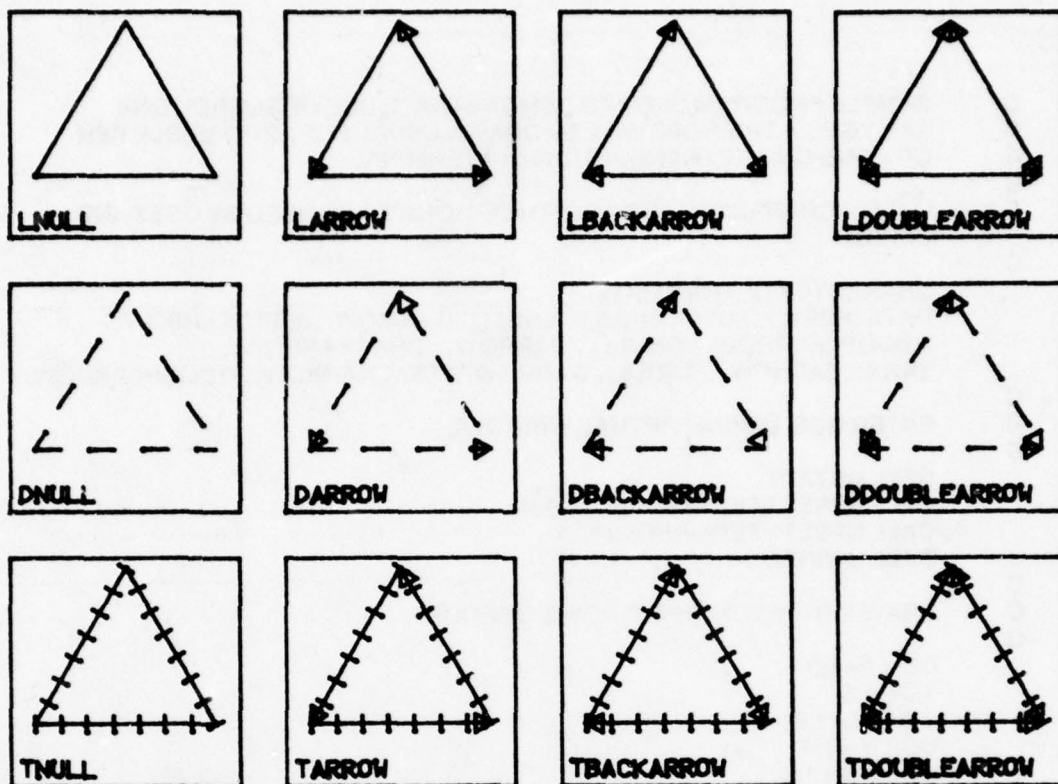
```

```

C   SAMPLE PROGRAM USED TO DEMONSTRATE USE OF SUBROUTINE
C   'UPLYGN'. A TRIANGLE WILL BE DRAWN USING 1 OF 12 POSSIBLE PEN
C   OPTIONS ON A TETRONIX 4010/4013 TERMINAL.
C
C   LOAD A CHARACTER ARRAY WITH OPTIONS TO BE USED BY USET AND
C   UPRINT
C
C   CHARACTER OPTION*16(12)
C   DATA INDEX,YO,OPTION/0.5,6,'LNULL;','LARROW;','LBACKARROW;
C   'LDOUBLEARROW;','DNULL;','DARROW;','DBACKARROW;
C   'DOUBLEARROW;','TNULL;','TARROW;','TBACKARROW;','TDOUBLARROW;/'
C
C   ENTER GCS, DEFINE VIRTUAL WINDOW.
C
C   CALL USTART
C   CALL UPSET ('TICINTERVAL', 0.25)
C   CALL UPSET ('TERMINATOR',';')
C   CALL UWINDO (-1.1,1.1,-1.1,1.1)
C
C   DRAW FIGURES IN THREE ROWS OF FOUR.
C
C   DO 1 I=1,3
C   XO=XO+1.5
C   YO=YO-1.8
C   DO 1 J=1,4
C   XO=XO+1.8
C   INDEX=INDEX+1
C
C   DEFINE DEVICE AREA SO THAT ONLY ONE POLYGON WILL BE DRAWN.
C
C   CALL UDAREA (XO,(XO+1.5),YO,(YO+1.5))
C   CALL UOUTLN
C
C   CALL 'UPLYGN' TO DRAW THE POLYGON WITHIN THE DEVICE AREA WE HAVE
C   DEFINED USING ONE OF THE TWELVE POSSIBLE PEN OPTIONS.
C
C   CALL USET (OPTION(INDEX))
C   CALL UPLYGN (0.,0.0,3.0,1.0)
C   CALL UPRINT (-1.0,-1.0,OPTION(INDEX))
1  CONTINUE
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VI-4



```

CHARACTER OPTION=16(12)
DATA INDEX, Y8/8, 5, 8/
DATA OPTION/'LNULL', 'LARROW', 'LBACKARROW', 'LDOUBLEARROW',
&          'DNULL', 'DARROW', 'DBACKARROW', 'DDOUBLEARROW',
&          'TNULL', 'TARROW', 'TBACKARROW', 'TDOUBLEARROW',
CALL USTART
CALL UPSET ('TERMINATOR', ',')
CALL UPSET ('TICINTERVAL', 8.26)
CALL UWINDO (-1.1, 1.1, -1.1, 1.1)
DO I I = 1, 3
  X8 = -1.5
  Y8 = Y8 - 1.8
  DO J J = 1, 4
    X8 = X8 + 1.8
    INDEX = INDEX + 1
    CALL UDAREA (X8, (X8+1.5), Y8, (Y8+1.5))
    CALL UOUTLN
    CALL USET (OPTION(INDEX))
    CALL UPLYON (8.8, 8.8, 3.8, 1.8)
    CALL UPRINT (-1.2, -1.8, OPTION(INDEX))
  I CONTINUE
CALL UEND
STOP
END

```

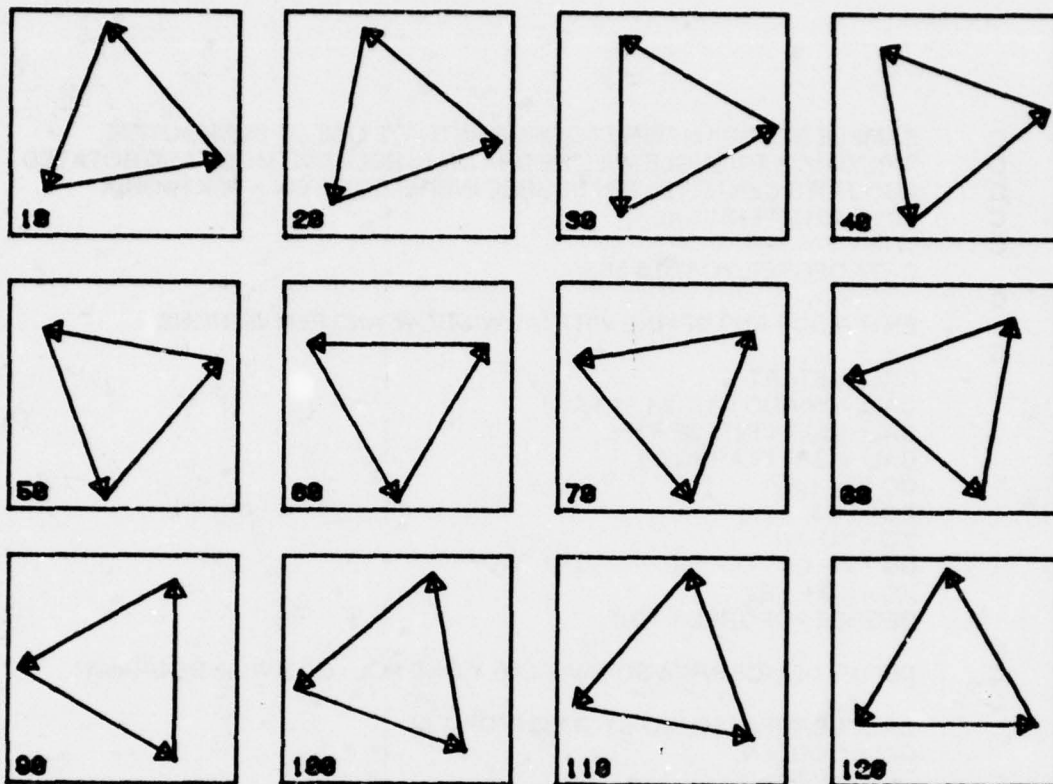


```

C   SAMPLE PROGRAM USED TO DEMONSTRATE USE OF SUBROUTINE
C   'UPLYGN'. A TRIANGLE WILL BE DRAWN IN RELATIVE MODE AND ROTATED
C   ABOUT ITS CENTER IN TEN DEGREE INCREMENTS ON A TEKTRONIX
C   4010/4013 TERMINAL.
C
C   DATA DEGREE, YO/0.0, 5.563/
C
C   ENTER GCS AND DEFINE VIRTUAL WINDOW, AND PEN OPTIONS
C
C   CALL USTART
C   CALL UWINDO (-1.1, 1.1, -1.1, 1.1)
C   CALL USET ('INTEGER')
C   CALL USET ('LARROW')
C   DO 1 I=1, 3
C   XO=-1.3
C   YO=YO-1.8
C   DO 1 J=1, 4
C   XO=XO+1.8
C   DEGREE=DEGREE+10.0
C
C   DEFINE DEVICE AREA SO THAT ONLY ONE POLYGON WILL BE DRAWN
C
C   CALL UDAREA (XO, (XO+1.5), YO, (YO+1.5))
C   CALL UOUTLN
C
C   CALL 'UPLYGN' TO DRAW THE POLYGON WITHIN THE DEVICE AREA WE HAVE
C   DEFINED. ROTATING IT IN RELATIVE MODE BY TEN DEGREES.
C
C   CALL UMOVE (0.0, 0.0)
C   CALL USET ('RELATIVE')
C   CALL UPSET ('ROTATE', DEGREE)
C   CALL UPLYGN (0.0, 0.0, 3.0, 1.0)
C   CALL USET ('ABSOLUTE')
C   CALL UPRINT (-1.0, -1.0, DEGREE)
1  CONTINUE
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VI-5



```

DATA DEGREE, Y8/8.8, 5.503/
CALL USTART
CALL UNDO (-1.1, 1.1, -1.1, 1.1)
CALL USET ('INTEGER')
CALL USET ('LARROW')
DO I = 1, 3
  X8 = -1.5
  Y8 = Y8 - 1.8
  DO J = 1, 4
    X8 = X8 + 1.8
  DEGREE = DEGREE + 18.8
  CALL UDAREA (X8, (X8+1.5), Y8, (Y8+1.5))
  CALL UOUTLN
  CALL UMOVE (8.8, 8.8)
  CALL USET ('RELATIVE')
  CALL USET ('ROTATE', DEGREE)
  CALL UPLYGN (8.8, 8.8, 3.8, 1.8)
  CALL USET ('ABSOLUTE')
  CALL UPRINT (-1.8, -1.8, DEGREE)
1 CONTINUE
CALL UEND
STOP
END

```

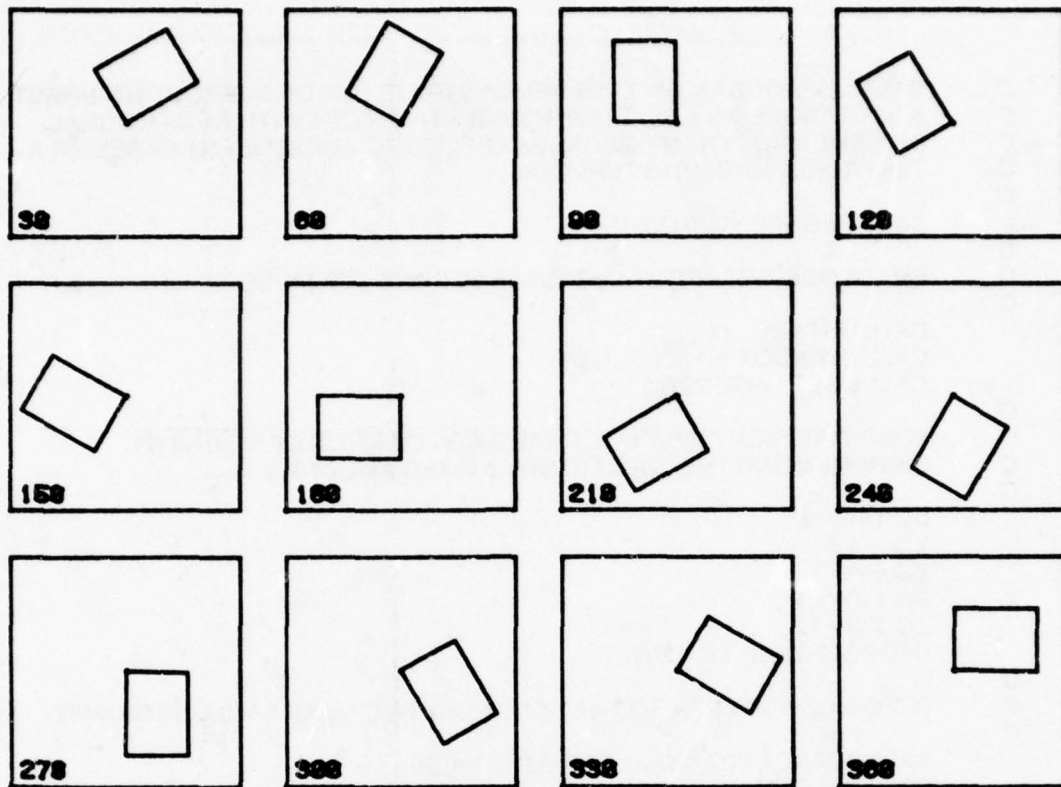
```

C   SAMPLE PROGRAM USED TO DEMONSTRATE USE OF SUBROUTINE 'URECT'.
C   A RECTANGLE WILL BE DRAWN IN RELATIVE MODE WITH A ROTATIONAL
C   INCREMENT OF THIRTY DEGREES APPLIED TO SUBSEQUENT CASES ON A
C   TEKTRONIX 4010/4013 TERMINAL.
C
C   DATA DEGREE,YO/0.0,5.63/
C
C   ENTER GCS, SET VIRTUAL WINDOW, SET INTEGER MODE
C
C   CALL USTART
C   CALL UWINDO (-1.1,1.1,-1.1,1.1)
C   CALL USET ('INTEGER')
C
C   LOOP THROUGH PROGRAM, CHANGING LOCATION OF FIGURE BY
C   INCREMENTING THE LOCATIO NBY A FIXED AMOUNT.
C
C   DO 1 I=1,3
C   XO=-1.5
C   YO=YO-1.8
C   DO 1 J=1,4
C   XO=XO+1.8
C   DEGREE=DEGREE+30.0
C
C   DEFINE DEVICE AREA SO THAT ONLY ONE RECTANGLE WILL BE DRAWN.
C
C   CALL UDAREA (XO,(XO+1.5),YO,(YO+1.5))
C   CALL UOUTLN
C
C   CALL 'URECT' TO DRAW THE RECTANGLE WITHIN THE DEVICE AREA WE
C   HAVE DEFINED, ROTATING IT IN RELATIVE MODE BY THIRTY DEGREES.
C
C   CALL UMOVE (0.0,0.0)
C   CALL USET ('RELATIVE')
C   CALL UPSET ('ROTATE',DEGREE)
C   CALL URECT (0.8,0.6)
C   CALL USET ('ABSOLUTE')
C   CALL UPRINT (-1.0,-1.0,DEGREE)
1  CONTINUE
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VI-6





```

DATA DEGREE, YB/8.8, 5.63/
CALL USTART
CALL UMINO (-1.1, 1.1, -1.1, 1.1)
CALL USET ('INTEGER')
DO I = 1, 3
  XB = -1.5
  YB = YB - 1.6
  DO J = 1, 4
    XB = XB + 1.6
    DEGREE = DEGREE + 36.8
    CALL UDAREA (XB, (XB+1.5), YB, (YB+1.5))
    CALL UOUTLN
    CALL UMOVE (8.8, 8.8)
    CALL USET ('RELATIVE')
    CALL UPSET ('ROTATE', DEGREE)
    CALL URECT (8.8, 8.8)
    CALL USET ('ABSOLUTE')
    CALL UPRINT (-1.8, -1.8, DEGREE)
  CONTINUE
CALL UEND
STOP
END

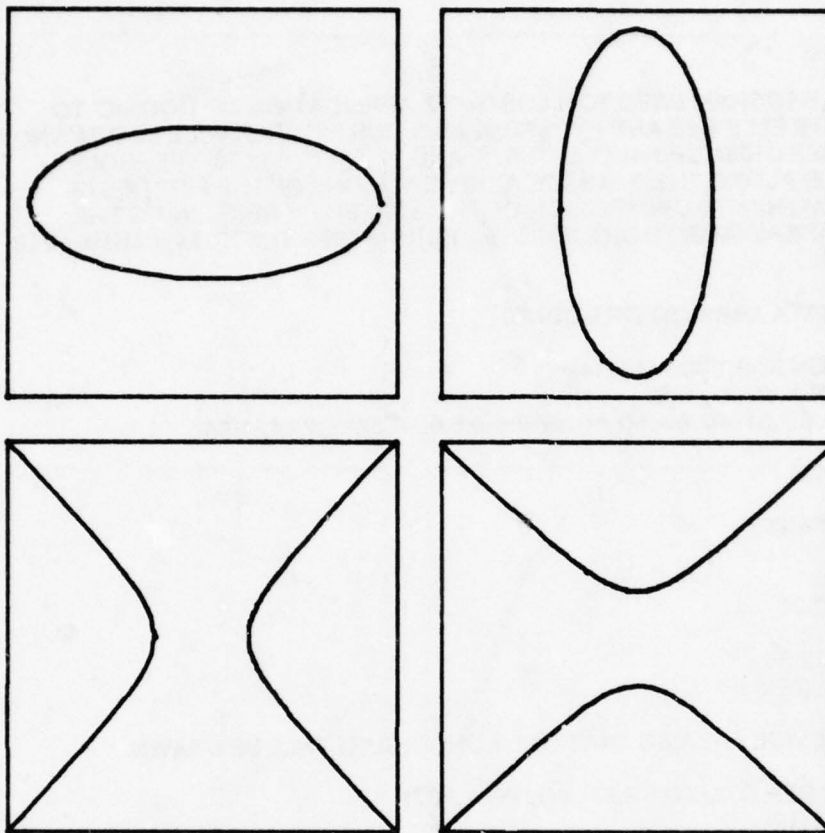
```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF 'UCONIC' TO
C   GENERATE ELLIPSES AND HYPERBOLAE. FOUR FIGURES WILL BE DRAWN:
C   2 ELLIPSES ORIENTED ALONG THE 'X' AND 'Y' AXES; and 2 HYPERBOLAE
C   ORIENTED ALONG THE 'X' AND 'Y' AXES. EACH ONE OF THE FIGURES IS
C   DRAWN WITHIN ITS OWN REGION OF THE SCREEN BY REDEFINING THE
C   DEVICE AREA PRIOR TO DRAWING THE FIGURE ON A TEKTRONIX 4010/4013
C   TERMINAL.
C
C   SET UP DATA ARRAYS FOR UCONIC
C
C   DIMENSION X(4),Y(4),P(4),E(4)
C   DATA INDEX,YO,X,Y,P,E/
C   0,5.73,10.,50.,67.,40.,50.,10.,50.,67,9.5-9.5,9.,-8.,9.,-9,1.44,-1.44/
C
C   ENTER GCS
C
C   CALL USTART
C   DO 1 I=1,2
C   XO=1.82
C   YO=YO-2.86
C   DO 1 J=1,2
C   XO=XO+2.86
C   INDEX=INDEX + 1
C
C   DEFINE DEVICE AREA SO THAT ONLY ONE FIGURE WILL BE DRAWN.
C
C   CALL UDAREA (XO,(XO+2.57),YO,(YO+2.57)
C   CALL UOUTLN
C
C   CALL UCONIC TO DRAW THE FIGURE USING THE PARAMETERS STORED IN
C   ARRAYS 'X','Y','P', AND 'E'. NOTE THAT DEFAULT VIRTUAL WINDOW IS
C   MAPPED TO THE CURRENT DEVICE AREA SPECIFICATION.
C
C   CALL UCONIC (X(INDEX),Y(INDEX),P(INDEX),E(INDEX),0.0,360.0)
1  CONTINUE
   CALL UEND
   STOP
   END

```

#### EXAMPLE VI-7



```

DIMENSION XC(4),YC(4),PC(4),EC(4)
DATA INDEX,Y8,X,Y,P,E/0,5.73,10.,50.,67.,50.,50.,10.,50.,67.,
2 9.5,-9.5,9.,-9.,.9,-.9,1.44,-1.44/
CALL USTART
DO 1 I = 1, 2
XB = -1.62
Y8 = Y8 - 2.66
DO 1 J = 1, 2
XB = XB + 2.66
INDEX = INDEX + 1
CALL UDAREA (XB,XB+2.57),Y8,(Y8+2.57)
CALL UOUTLN
CALL UCONIC (X(INDEX),Y(INDEX),P(INDEX),E(INDEX),0.0,360.0)
1 CONTINUE
CALL UEND
STOP
END

```

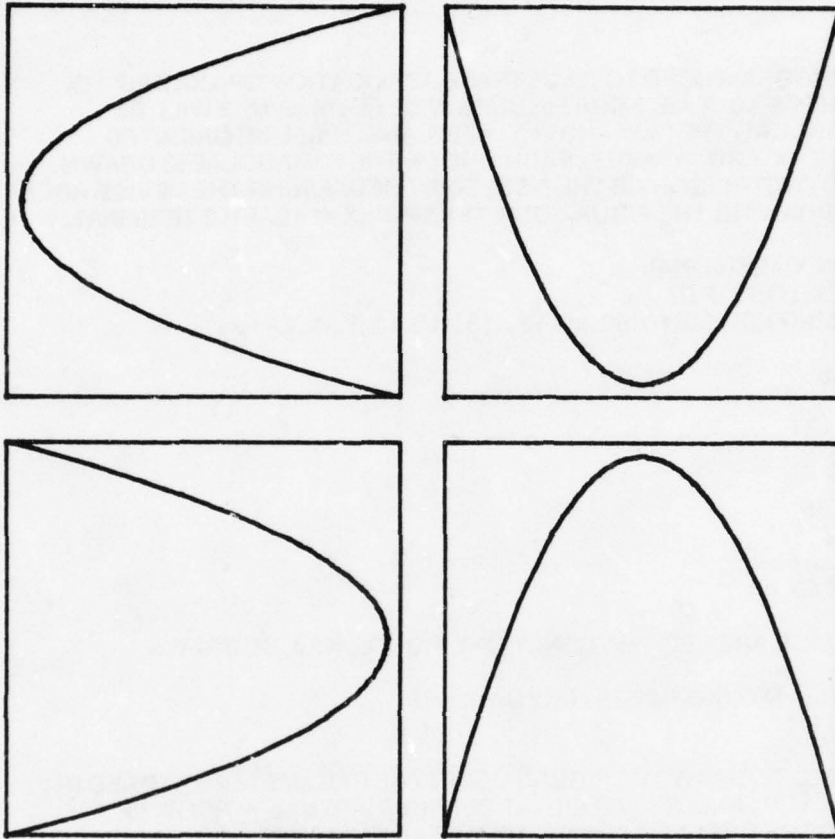


```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF 'UCONIC' TO
C   GENERATE PARABOLAE. FOUR FIGURES WILL BE DRAWN: 2 WILL BE
C   ORIENTED ALONG THE '+X' AND '+Y' AXES, AND 2 WILL BE ORIENTED
C   ALONG THE '-X' AND '-Y' AXES. EACH ONE OF THE PARABOLAE IS DRAWN
C   WITHIN ITS OWN REIGON OF THE SCREEN BY REDEFINING THE DEVICE AREA
C   PRIOR TO DRAWING THE FIGURE ON A TEKTRONIX 4010/4013 TERMINAL.
C
C   DIMENSION X(4),Y(4),P(4)
C   DATA INDEX,YO,X,Y,P,E/
C   0,5.73,10.,50.,90.,50.,50.,10.,50.,90.,13.,-13.,-13.,13.,1.,-1.,1.,-1./
C
C   ENTER GCS
C
C   CALL USTART
C   DO 1 I=1,2
C   XO=1.82
C   YO=YO-2.86
C   DO 1 J=1,2
C   XO=XO+2.86
C   INDEX=INDEX+1
C
C   DEFINE DEVICE AREA SO THAT ONLY ONE FIGURE WILL BE DRAWN.
C
C   CALL UDAREA (XO,(XO+2.57),YO,(YO+2.57))
C   CALL UOUTLN
C
C   CALL UCONIC TO DRAW THE FIGURE USING THE PARAMETERS STORED IN
C   ARRAYS 'X','Y','P', AND 'E'. NOTE THAT DEFAULT VIRTUAL WINDOW IS
C   MAPPED TO THE CURRENT DEVICE AREA SPECIFICATION.
C
C   CALL UCONIC (X(INDEX),Y(INDEX),P(INDEX),E(INDEX),0.0,360.)
1  CONTINUE
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VI-8



```

DIMENSION X(4),Y(4),P(4),E(4)
DATA INDEX,Y0,X,Y,P,E/8,5.73,10.,50.,80.,50.,50.,10.,50.,80.,
2      13.,-13.,-13.,13.,1.,-1.,1.,-1./
CALL USTART
DO 1 I = 1, 2
  X0 = -1.82
  Y0 = Y0 - 2.80
  DO 1 J = 1, 2
    X0 = X0 + 2.80
    INDEX = INDEX + 1
    CALL UDAREA (X0,X0+2.57),Y0,(Y0+2.57))
    CALL UOUTLN
    CALL UCONIC (X(INDEX),Y(INDEX),P(INDEX),E(INDEX),8.8,300.8)
1  CONTINUE
    CALL UEND
    STOP
    END

```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF 'ULINFT' TO
C   CALCULATE THE SLOPE (S) AND Y-INTERCEPT (YI) OF A LINE WHICH
C   REPRESENTS THE 'BEST' LINEAR FIT TO A SERIES OF DATA POINTS.
C   DEFAULT VALUES OF VIRTUAL WINDOW, DEVICE AREA, AND LINE TYPE ARE
C   USED.
C
C   DIMENSION X(20), Y(20)
C   DATA X/0.,5., 10., 15., 20., 25., 30., 35., 40., 45., 50., 55., 60., 65., 70., 75., 80., 85.,
1  90., 95./
C   DATA Y/20., 25., 30., 35., 30., 25., 20., 15., 10., 15., 20., 25., 30., 35., 40., 45., 50., 55.,
1  60., 65./
C
C   ENTER GCS, DRAW OUTLINE
C
C   CALL USTART
C   CALL UOUTLN
C   CALL USET ('N+')
C   DO 2 I=1,20
C
C   AND PLOT EACH DATA POINT WITH A '+'.
C
C   CALL UPEN (X(I),Y(I))
1  CONTINUE
C   CALL USET ('LINE')
C
C   CALL 'ULINFT' TO CALCULATE THE LINE'S SLOPE AND Y-INTERCEPT.
C
2  CALL ULINFT (X,Y,20,S,YI)
C
C   MOVE TO THE Y-INTERCEPT THEN GRAPH THE LINE USING  $YO = SX + YI$ .
C
C   XMIN=0.0
C   XMAX=100.0
C   CALL UMOVE (XMIN,YI)
C   YO=YI+S*XMAX
C   CALL UPEN (XMAX,YO)
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VI-9



AD-A063 167

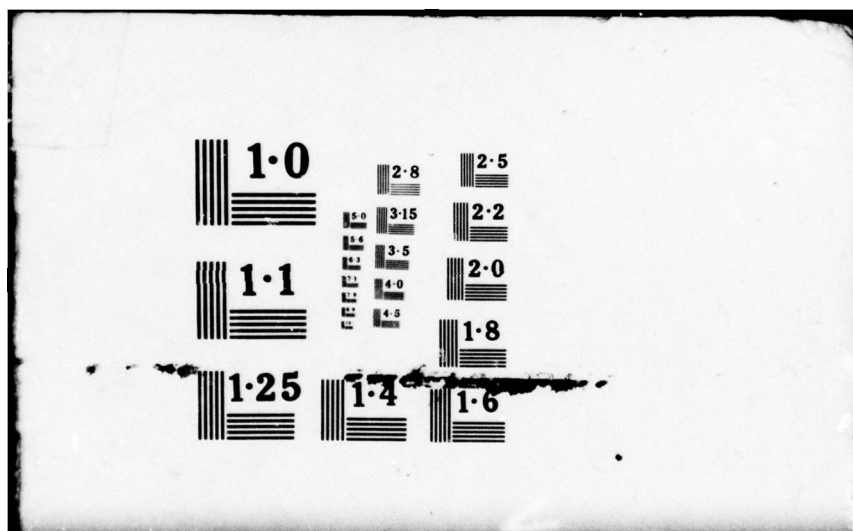
ARMY ENGINEER WATERWAYS EXPERIMENT STATION VICKSBURG MISS. F/G 9/2  
GRAPHICS COMPATIBILITY SYSTEM (GCS). PRIMER ON COMPUTER GRAPHIC--ETC(U)  
1978

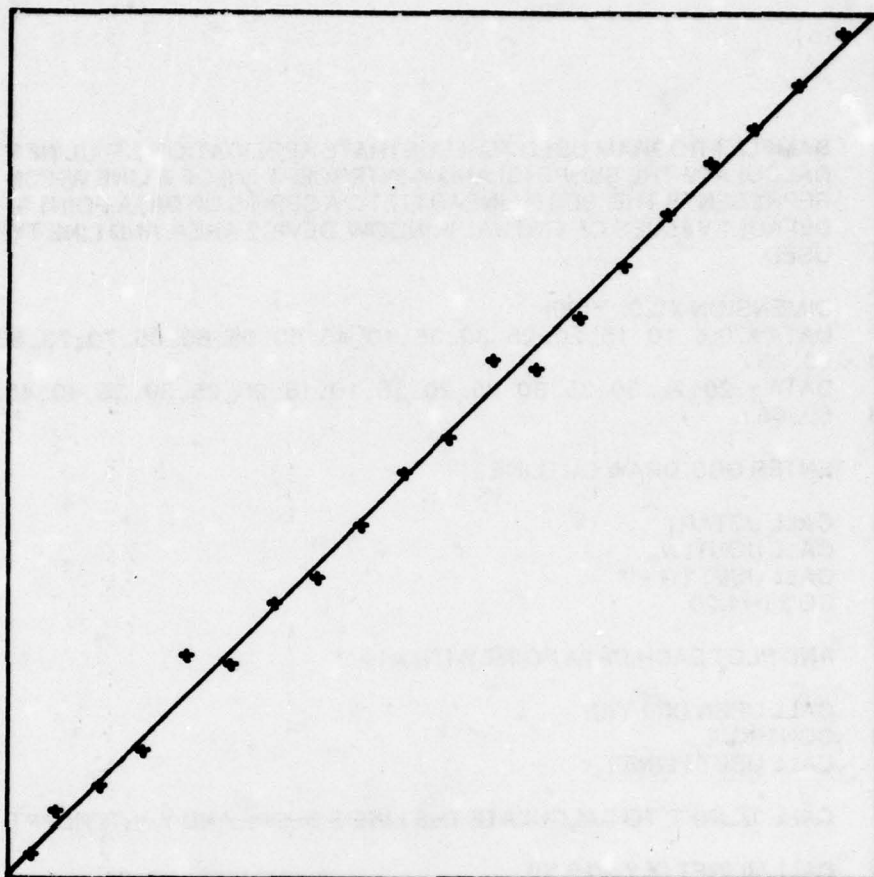
UNCLASSIFIED

NL

2 OF 5  
AD A  
063167







```

DIMENSION X(20),Y(20)
DATA X/2.,5.,10.,15.,20.,25.,30.,35.,40.,45.,
      50.,55.,60.,65.,70.,75.,80.,85.,90.,95./
DATA Y/2.,7.,10.,14.,25.,24.,31.,34.,40.,40.,
      50.,50.,50.,64.,70.,70.,62.,60.,91.,97./
CALL USTART
CALL UOUTLN
DO 2 I = 1, 20
  XN = FLOAT(I)
  CALL USET ('ACENTER')
  CALL USET ('N+')
  CALL UPEN (X(I),Y(I))
  CALL USET ('LINE')
2 CONTINUE
CALL ULINFT (X,Y,XN,S,YI)
XMIN = 0.0
XMAX = 100.0
CALL UNOVE (XMIN,YI)
YB = YI + S * XMAX
CALL UPEN (XMAX,YB)
CALL UEND
STOP
END

```



```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF 'ULSTSQ' TO
C   CALCULATE THE COEFFICIENTS OF A POLYNOMIAL OF ORDER IDEGRE-1
C   WHICH REPRESENTS THE 'BEST' FIT TO A SERIES OF DATA POINTS.
C   DEFAULT VALUES OF VIRTUAL WINDOW, DEVICE AREA, AND LINE TYPE ARE
C   USED. EACH OF THE DATA POINTS ARE PLOTTED WITH A 'I'. AFTER
C   'ULSTSQ' IS CALLED, THE POLYNOMIAL IS THEN GRAPHED, USING THE
C   COEFFICIENTS WHICH WERE COMPUTED.

    PARAMETER IDEGRE=7
    DIMENSION A(IDEGRE),X(20),Y(20)
    DATA X/0.,5., 10., 15., 20., 25., 30., 35., 40., 45., 50., 55., 60., 65., 70., 75., 80., 85.,
1    90., 95./
    DATA Y/20., 25., 30., 35., 30., 25., 20., 15., 10., 15., 20., 25., 30., 35., 40., 45., 50.,
2    55., 60., 65./

C   ENTER GCS, DRAW OUTLINE, SET TYPE AND DEGREE OF FIT
C
    CALL USTART
    CALL UOUTLN
    CALL UPSET ('POLYNOMIAL',LFOAT(IDEGRE-1))
    CALL USET ('N+')

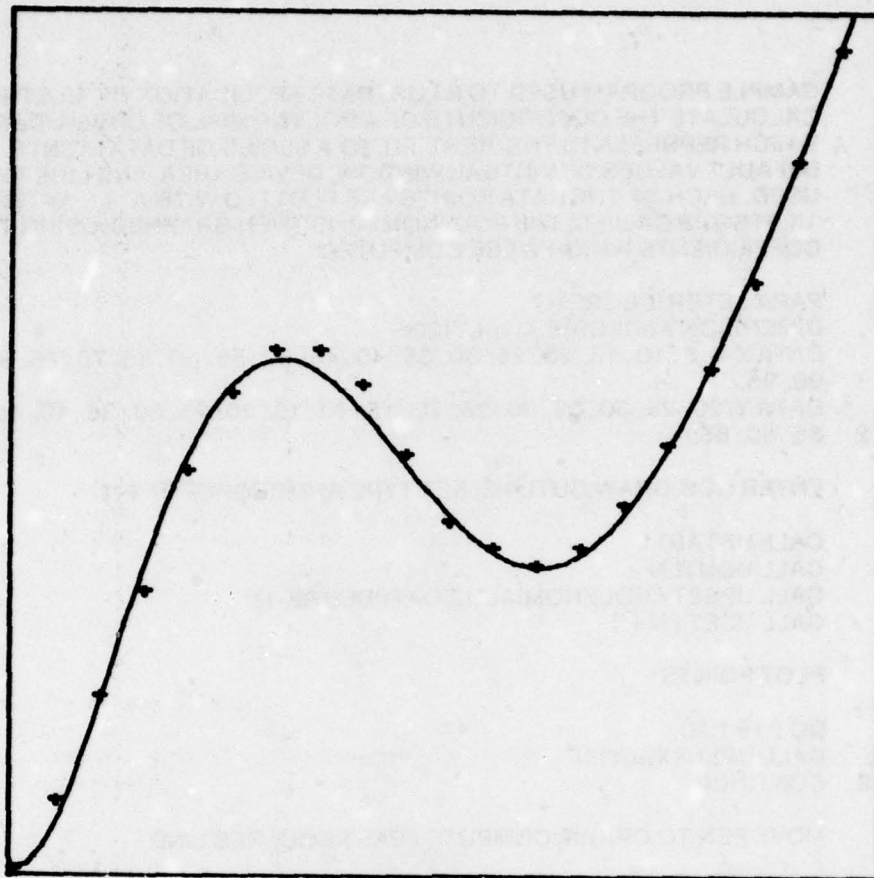
C   PLOT POINTS
C
    DO 2 I=1,20
      CALL UPEN (X(I),Y(I))
2    CONTINUE

C   MOVE PEN TO ORIGIN, COMPUTE LEAST SQUARES LINE
C
2    CALL UMOVE
    CALL ULSTSQ (X,Y,20.,A)

C   PLOT LEAST SQUARES LINE APPROXIMATING POINTS
C
    DO 5 I=1,100
      YO=A(1)
      XO=FLOAT(1)
      XK=XO
      DO 4 J=2, IDEGREE
        YO=A(J)*XK+YO
        XK=XK*XO
3      CONTINUE
      CALL UPEN (XO,YO)
4      CONTINUE
      CALL UEND
      STOP
    END

```

#### EXAMPLE VI-10



```

PARAMETER IDEGRE=7
DIMENSION A(IDEGRE),X(20),Y(20)
DATA X/0.,5.,10.,15.,20.,25.,30.,35.,40.,45.,
      50.,55.,60.,65.,70.,75.,80.,85.,85./
DATA Y/0.,8.,29.,32.,40.,55.,68.,68.,50.,48.,
      40.,37.,35.,37.,42.,40.,58.,68.,82.,85./
CALL USTART
CALL UOUTLN
CALL USET ('POLYNOMIAL',FLOAT(IDEGRE-1))
CALL USET ('ACENTER')
CALL USET ('N+')
DO 2 I = 1, 20
CALL UPEN (X(I),Y(I))
2 CONTINUE
CALL USET ('LINE')
CALL UMOVE (0,0,0,0)
CALL ULST00 (X,Y,20.,A)
DO 5 I = 1, 100
YB = A(I)
XB = FLOAT(I)
XK = XB
DO 4 J = 2, IDEGRE
YB = A(J) * XK + YB
XK = XK * XB
4 CONTINUE
CALL UPEN (XB,YB)
5 CONTINUE
CALL UEND
STOP
END

```

## CHAPTER VII

### HIGH LEVEL GRAPHICS

#### High Level Plotting Facility

UPLOT is a general-purpose composite plotting subroutine which may be used to graph full-screen representations of complete arrays of data. Through the use of common default options, UPLOT provides sufficient flexibility to meet the majority of plotting requirements; however, should the user desire to change any of the applicable options, UPLOT will permit:

- A. Rectangular, semi-logarithmic, log-log, or polar plotting;
- B. Automatic or forced scaling of data values;
- C. Axes presentation and format options, to include alphanumeric labeling.

UPLOT may be called by the following sequence:

CALL UPLOT (X,Y,CURVES, PARRAY,OPTARY)

CURVES is a single-valued variable which denotes the number of curves to be plotted. PARRAY is an array which describes the number of data points for each curve; e.g., PARRAY(1) is the number of data points comprising the first curve, PARRAY(2) is the number of points for the second curve, etc. OPTARY is a character array which specifies the data presentation format applicable to each given curve. Each element of OPTARY must not exceed 4 characters in length, and must represent a valid pen-status option such as 'LINE', 'VECT', etc. Since PARRAY and OPTARY follow the same general nomenclature in describing the characteristics of a given curve, both of these arrays must have as many elements as there are curves to be plotted. Arrays X and Y contain the actual data values grouped by curve. The size of the X and Y arrays must be equal to the sum of the elements of PARRAY. The general format of each of these arrays is outlined in the following figure.

#### Contents of Arrays Used in UPLOT

PURPOSE OF PLOT - Plot three curves having 8, 4 and 5 points each

	X	Y	CURVES	PARAY	OPTARY
Curve 1	X <sub>11</sub>	Y <sub>11</sub>	3.	8.	'LINE'
	X <sub>12</sub>	Y <sub>12</sub>		4.	'VECT'
	X <sub>13</sub>	Y <sub>13</sub>		5.	'DARR'
	X <sub>14</sub>	Y <sub>14</sub>			
	X <sub>15</sub>	Y <sub>15</sub>			
	X <sub>16</sub>	Y <sub>16</sub>			
	X <sub>17</sub>	Y <sub>17</sub>			
	X <sub>18</sub>	Y <sub>18</sub>			
Curve 2	X <sub>21</sub>	Y <sub>21</sub>			
	X <sub>22</sub>	Y <sub>22</sub>			
	X <sub>23</sub>	Y <sub>23</sub>			
	X <sub>24</sub>	Y <sub>24</sub>			



Curve 3

X<sub>31</sub>  
X<sub>32</sub>  
X<sub>33</sub>  
X<sub>34</sub>  
X<sub>35</sub>

Y<sub>31</sub>  
Y<sub>32</sub>  
Y<sub>33</sub>  
Y<sub>34</sub>  
Y<sub>35</sub>

*For those users desiring to plot only a single curve using the current line type, there exists a special form of UPLLOT — UPLLOT1 — which has a simplified calling sequence. UPLLOT1 offers all of the various options available through UPLLOT and may be called by the following sequence:*

**CALL UPLLOT1 (X,Y,PTS)**

*X and Y are arrays containing the data values to be plotted and PTS is a single-valued variable indicating the number of points which comprise the curve.*

*The visual results obtained through UPLLOT depend upon the status of the GSA at the time UPLLOT is invoked. All of the high-level graphics parameters are assigned default values by USTART, and may be modified through conventional calls to USET and UPSET. Although many of these options will be presented, it is not the intent of this chapter to discuss each particular option in detail, further information may be found in the subroutine writeups*

#### **Coordinate System Options**

The two types of coordinate systems applicable to UPLLOT are specified through USET in the following manner:

**CALL USET ('RECTANGULAR')  
CALL USET ('POLAR')**

Under RECTANGULAR coordinates, the user is given the option of specifying linear or logarithmic mappings of the independent and dependent variables: 'LINXAXIS', 'LOGXAXIS', 'LNXAXIS', 'LINYAXIS', 'LOGYAXIS', and 'LNYAXIS'. The 'LN' axis option implies that natural (Napierian) logarithms, as opposed to common (base 10) logarithmic transformations are to be applied to the data values when they are displayed; this option is particularly useful for Arrhenius plots and other applications in which the variables are related by an exponent of 'e'.

#### **Scaling Options**

There are three options applicable to the scaling of data values for a graphical display under UPLLOT: 'AUTOSCALE', 'FULLSCALE', and 'OWNSCALE'. Under AUTOSCALE, the range of values to be plotted is examined and scaling occurs so that the graph will be presented with 'nice' numbers at the tic marks on the axis (if requested). The FULLSCALE option resembles AUTOSCALE with the exception that a zero is forced on the X-axis. OWNSCALE uses the virtual window established by the user upon entry to UPLLOT in order to scale the data values, hence, a zero-value is not forced on the X-axis. Thus, 'nice' numbers may not necessarily appear as numeric labels, and all of the data points may not be displayed should they extend beyond the window limits.

There are also two options relating to the calculation of the scale factors, 'NEWSCALE' and 'OLDSCALE'. With the 'NEWSCALE' option, the scale factors used to map the data onto the display screen is always recalculated. Specifying 'OLDSCALE' results in this calculation being bypassed, with the result that a scaling factor used in a previous plot is



used for the current plot. This offer results in an increase in speed at which the plots are produced.

### **General Purpose Axes Creation**

Subroutine **UAXIS** is invoked by subroutine **UPLOT** to generate the desired axes for the input data arrays. It is specified:

**CALL UAXIS(XMIN,XMAX,YMIN,YMAX)**

### **Axes Options**

The options which apply to the axes may be further subdivided into the following categories:

- A. Axes existence options.
- B. Axes format options.
- C. Axes positioning options
- D. X and Y labeling option.
- E. Numeric label format options.

'**XYAXES**', '**XAXIS**', '**YAXIS**', and '**NOAXES**' comprise the USET options which specify the existence of an axis. XYAXES indicates that both X and Y axes are to be drawn, whereas the XAXIS or YAXIS options are be used when only one of the respective axes is desired. The NOAXES option will suppress the display of the X and Y axes, but will permit axis labelling should the user so desire. The existence or lack of any axis does not influence the scaling mode which the user has requested.

The axes format options consist of allowing the user to specify either '**PLAINAXES**', '**TICAXES**', or '**GRIDAXES**'. Under PLAINAXES, the specified axis (or axes) will appear as a solid line. The TICAXES option will force the axes to be drawn as ticked lines, with the tic intervals specified by the UPSET options:

**CALL UPSET ('TICX',XINTERVAL)**  
**CALL UPSET ('TICY',YINTERVAL)**

XINTERVAL and YINTERVAL are single-valued REAL variables which specify the interval (in current units) at which tic marks are to appear on the X and Y axes respectively. Under the default values of zero X and Y tick intervals, UPLOT will choose suitable intervals based upon the axes existence and data scaling options specified by the user. The GRIDAXES option should be used whenever grid lines are desired for the axes which have been defined. The grid spacing will be determined in the same manner as the tic interval under TICAXES. Under POLAR coordinates, XINTERVAL defines the radial distance between concentric circles which comprise the grid lines for the R-axis; whereas, YINTERVAL specifies the angular increment at which grid lines which be drawn for the O-axis.

The user may position his axes through the use of the following options: '**EDGEAXES**', '**ZEROAXES**', '**XZEROYEDGE**', '**YEDGEYZERO**', '**XEDGEYZERO**', and '**YZEROXEDGE**'. The EDGEAXES option indicates that the axis (or axes) is to be oriented near the edges of the currently defined device area. When the EDGEAXIS option is used in conjunction with XYAXES and OWNSCALE, UPLOT will generate X and Y axes which intersect at the

minimum values contained within the X and Y arrays. In addition, the axes will be positioned near the left and bottom edges of the device area. Should the minimum and maximum values of the X and Y arrays constitute an interval which contains zero, the ZEROAXES option may be used to force the intersection of the axes through the given zero point. Should ZEROAXES be specified, and zero is not contained within the defined interval, EDGEAXES will be generated. XZEROYEDGE, YEDGEZERO, XEDGEZERO, and YZEROEDGE options are merely extensions of the EDGEAXES and ZEROAXES options, and are subject to the constraints outlined above.

The user is provided with the ability to control the format under which the X and Y axes are labeled through the use of the following options: 'NOXLABEL', 'XNUMERICLABEL', 'XALPHALABEL', 'XBOTHLABELS', 'NOYLABEL', 'YNUMERICLABEL', 'YALPHALABEL', and 'YBOTHLABELS'. 'NOXLABEL' and 'NOYLABEL' should be used when the labeling of the X and Y axes is to be suppressed. The 'XNUMERICLABEL' and 'YNUMERICLABEL' options indicate that numeric labels are to be created along the X and Y axes at the tic intervals specified by 'TICX' and 'TICY' respectively. The actual values for the numeric labels will be based upon the axes scaling options previously discussed. 'XALPHANUMERIC' and 'YALPHANUMERIC' allow the user to generate alphanumeric labels (or titles) for each axis. The actual information to be displayed is defined through a call to UPSET:

**CALL UPSET ('XLABEL',XDATA)  
CALL UPSET ('YLABEL',YDATA)**

XDATA and YDATA are strings of up to 40 characters in length, with the standard string termination character inserted at the end of the information. The XBOTHLABELS and YBOTHLABELS options are used when numeric as well as alphabetic labels are desired.

The user is provided with the ability to control the format under which numeric labels will be printed through the use of the 'BESTFORMAT', 'IFORMAT', and 'GFORMAT' options. Under 'IFORMAT', all numeric labels will appear as integers, with truncation occurring for any non-integral values. GFORMAT will generate labels which will be edited into either standard FORTRAN 'E' or 'F' formats. BESTFORMAT implies that the numeric labels are to be output under the format which is most appropriate for the given label.

#### **Time Series Axes**

In addition, there is the axes routine, UTAXIS, that permits the user to select, scale and label axes. For further information on this routine, see the subroutine description section. It is specified:

**CALL UTAXIS(BEGPER,PERIOD,YMIN,YMAX)**

#### **Curve Fitting Options**

UPLLOT provides an additional facility for those users desiring to fit linear, least-squares polynomial, and spline curves to their data points. Interested users are directed to the detailed descriptions of UPLLOT, ULSTSQ, and ULINFT contained within the subroutine description section for further details of this capability.

#### **Extended High-Level Graphics**

For data presentation formats extending beyond those available through UPLLOT and UPLLOT1, a series of generalized histogram, barchart and piechart utility subroutines have been incorporated into GCS. These subroutines are comprehensive enough to permit their utilization as the core of a complete data analysis and display program;



without compromising the versatility to be incorporated into a graphics program within which they play a relatively minor role.

Histogram generation facilities are provided through the use of subroutine UHISTO:

**CALL UHISTO (DATA,XN,CELLS)**

DATA is a REAL array containing the data values from which the histogram is to be constructed, XN is a single-valued REAL variable which is used to specify the number of elements in the DATA array; and CELLS is a single-valued REAL variable which designates the number of cells (or bars) which are to appear on the histogram. The standard axes existence, labeling and scaling options previously discussed for UPLOT also apply to UHISTO, with the exception of the GRIDAXIS option for the X and Y axes, and 'TICAXIS' for the Y axis.

Subroutine UBAR may be invoked whenever the user desires to generate a barchart.

**CALL UBAR (DATA,XN,LABELS,SLABEL)**

DATA and XN are interpreted in the same manner as described above the UHISTO. LABELS is a character array which is used to specify the label for each XN values stored in DATA. SLABEL is a single-valued REAL variable which defines the length (in characters) of each element of the LABELS array. Should UBAR encounter the standard string termination character (:) while processing any of the label items, no additional characters are obtained from the item and the string prior to the terminator is scaled to a string of SLABEL characters in length.

Subroutine UCHART may be invoked whenever the user desires to generate a grouped bar chart for multi-valued data.

**CALL UCHART (ARRAY,GROUPS,BARS,LABELS,YMAXL)**

Standard piechart generation facilities are available under GCS through subroutine UPIE which may be called by the following sequence:

**CALL UPIE (DATA,XN,LABELS,SLABEL)**

DATA, XN LABELS and SLABEL are interpreted in the same manner as under UBAR.

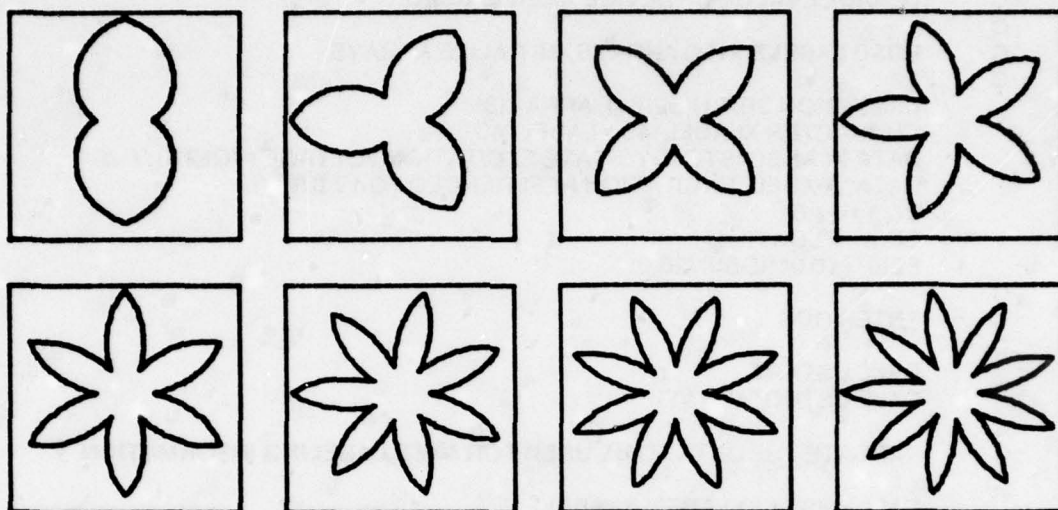


```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF SUBROUTINE
C   'UPLOT1'. EIGHT CURVES WILL BE PLOTTED IN POLAR COORDINATES,
C   USING THE SINGLE-CURVE POLAR PLOTTING ROUTINE. THE PURPOSE OF
C   THIS EXAMPLE IS TO ILLUSTRATE THE USE OF THE HIGH-LEVEL
C   SUBROUTINE TO PLOT SINGLE CURVES WITHIN THE USER-DEFINED DEVICE
C   AREA. NO AXES OR AXES LABELING IS REQUESTED. THIS WAS WRITTEN
C   FOR A TEKTRONIX 4010/4013 TERMINAL.
C
C   INITIALIZE CONSTANTS AND ANGLES
C
C   DIMENSION R(361), THETA(361)
C   DATA PI,W,YO/3.1415925,0.0,4.71/
C   DO 1 I=1,361
1  THETA(I)=FLOAT(I)*PI/180.0
C
C   ENTER GCS, SET FOR POLAR, RADIAN MEASURE, NO AXES AND NO LABELS.
C
C   CALL USTART
C   CALL USET ('POLAR')
C   CALL USET ('RADIAN')
C   CALL USET ('NOAXES')
C   CALL USET ('NOXLABEL')
C
C   INCREMENT LOCATION OF WORKING AREA, AND PLOT THE FIGURES.
C
C   DO 3 I=1,2
C   XO=-1.5
C   YO=YO-1.8
C   DO 3 J=1,4
C   W=W+0.5
C   XO=XO+1.8
C   CALL UDAREA (XC,(XO+1.5),YO,(YO+1.5))
C   CALL UOUTLN
C   DO 2 K=1,361
2  R(K)=1.2-0.7*(ABS(COS(W*THETA(K)))-ABS(SIN(W*THETA(K))))
C
C   PLOT 361 DATA POINTS SPECIFIED BY 'R' AND 'THETA' USING THE
C   STANDARD LINE OPTION WITHIN THE DEVICE AREA WE HAVE DEFINED.
C   NOTE THAT NO AXES OR AXES LABELING WAS SPECIFIED.
C
C   CALL UPLOT1 (R,THETA,361.0)
3  CONTINUE
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VII-1



```

DIMENSION R(361), THETA(361)
DATA PI, W, YB/3.14159265, 8.5, 4.71/
DO 1 I = 1, 361
1 THETA(I) = FLOAT(I) * PI / 180.8
CALL USTART
CALL USET ('POLAR')
CALL USET ('RADIANS')
CALL USET ('NOAXES')
CALL USET ('NOXLABEL')
CALL USET ('NOYLABEL')
DO 3 I = 1, 2
XB = -1.5
YB = YB - 1.8
DO 3 J = 1, 4
W = W + 8.5
XB = XB + 1.8
CALL UDAREA (XB, XB+1.5, YB, (YB+1.5))
CALL UOUTLN
DO 2 K = 1, 361
2 R(K) = 1.2-0.7*(ABS(COS(W*THETA(K)))-ABS(SIN(W*THETA(K))))
CALL UPLOTT (R, THETA, 361.8)
3 CONTINUE
CALL UEND
STOP
END

```

```

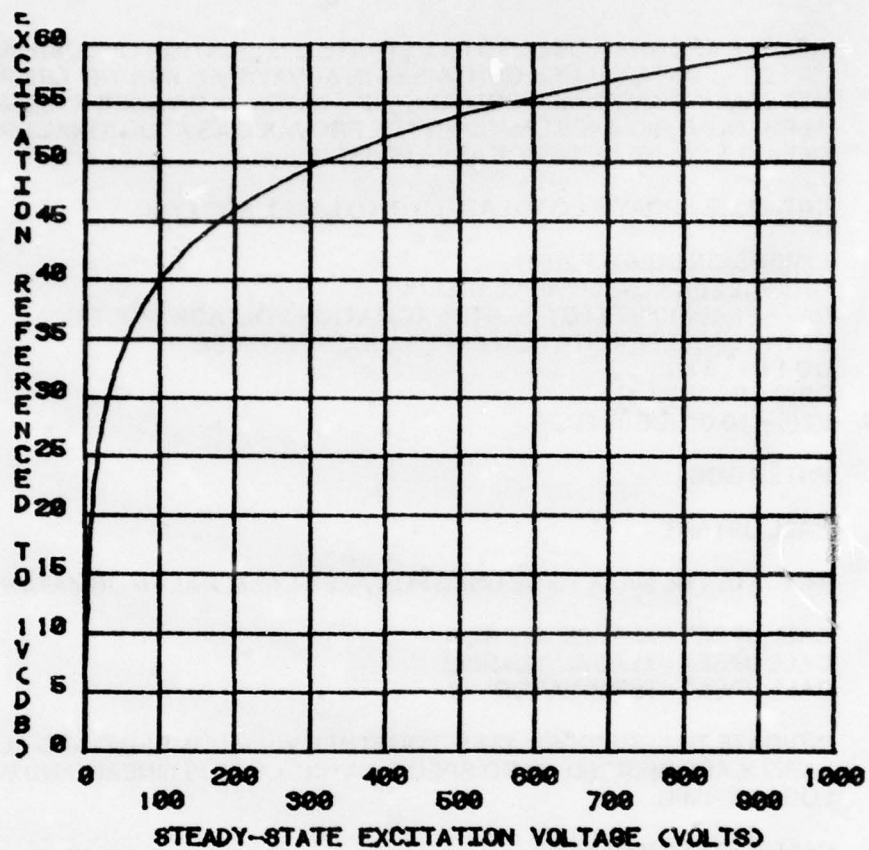
C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF SUBROUTINE
C   'UPLOT1'. DATA VALUES CONTAINED IN ARRAYS 'E2' AND 'DB' ARE PLOTTED
C   IN A CARTESIAN COORDINATE SYSTEM. A GRIDDED AXES, AND
C   ALPHANUMERIC AXES LABELING ARE PROVIDED AS ADDITIONAL OPTIONS.
C   DEFAULT VALUE OF DEVICE AREA IS USED.
C
C   LOAD LABELS INTO ARRAYS, INITIALIZE ARRAYS.
C
C   DIMENSION DB(61),E2(61), ARRAY(8)
C   CHARACTER XLABEL*40,YLABEL*40
C   DATA XLABEL/'STEADY-STATE EXCITATION VOLTAGE (VOLTS)';/
C   DATA YLABEL/'EXCITATION REFERENCED TO 1V (DB)';/
C   DO 1 I=1,61
C     DB(I)=FLOAT(I-1)
C     E2(I)=10.0*(DB(I)/20.0)
1
C
C   ENTER GCS
C
C   CALL USTART
C   CALL USTUD (ARRAY)
C
C   INDICATE THE DATA TO BE USED FOR AXES LABELING INFORMATION.
C
C   CALL UPSET ('XLABEL',XLABEL)
C   CALL UPSET ('YLABEL',YLABEL)
C   CALL UPSET ('TERMINATOR',':')
C
C   INDICATE THAT GRIDDED AXES, TOGETHER WITH NUMERIC AND ALPHA
C   LABELS ARE DESIRED.
C
C   CALL USET ('GRIDAXES')
C   CALL USET ('XBOTHLABEL')
C   CALL USET ('YBOTHLABEL')
C
C   CALL 'UPLOT1' TO PLOT A SINGLE CURVE OF 61 POINTS USING 'E2' AND 'DB'.
C   STANDARD LINE OPTION IS REQUESTED.
C
C   CALL UPLOT1 (E2,DB,61.0)
C
C   USE CROSSHAIRS TO DEFINE LOWER BOUNDARY THEN UPPER BOUNDARY.
C   THEN CALL UWINDO AND PLOT JUST THAT PORTION.
C
C   CALL UGRIN (XL,YL,IC)
C   CALL UGRIN (XU,YU,IC)
C   CALL UWINDO (XL,XI,YL,YU)
C   CALL USET ('OWNSCALE')
C   CALL UDAREA (ARRAY(5), ARRAY(6), ARRAY(7), ARRAY(8))
C   CALL UERASE
C   CALL UPLOT1 (E2,DB,61.)
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VII-2

*vil - 02*





```

DIMENSION DB(61), E2(61)
CHARACTER XLABEL*40, YLABEL*40
DATA XLABEL/'STEADY-STATE EXCITATION VOLTAGE (VOLTS)', '/'
DATA YLABEL/'EXCITATION REFERENCED TO IV(DB)', '/'
DO 1 I = 1, 61
  DB(I) = FLOAT(I-1)
1 E2(I) = 10.0*(DB(I) / 20.0)
CALL USTART
CALL UPSET ('TERMINATOR', ',')
CALL UPSET ('XLABEL', XLABEL)
CALL UPSET ('YLABEL', YLABEL)
CALL USET ('GRIDAXES')
CALL USET ('XBOTH LABEL')
CALL USET ('YBOTH LABEL')
CALL UPLOT1 (E2, DB, 61.0)
CALL UEND
STOP
END

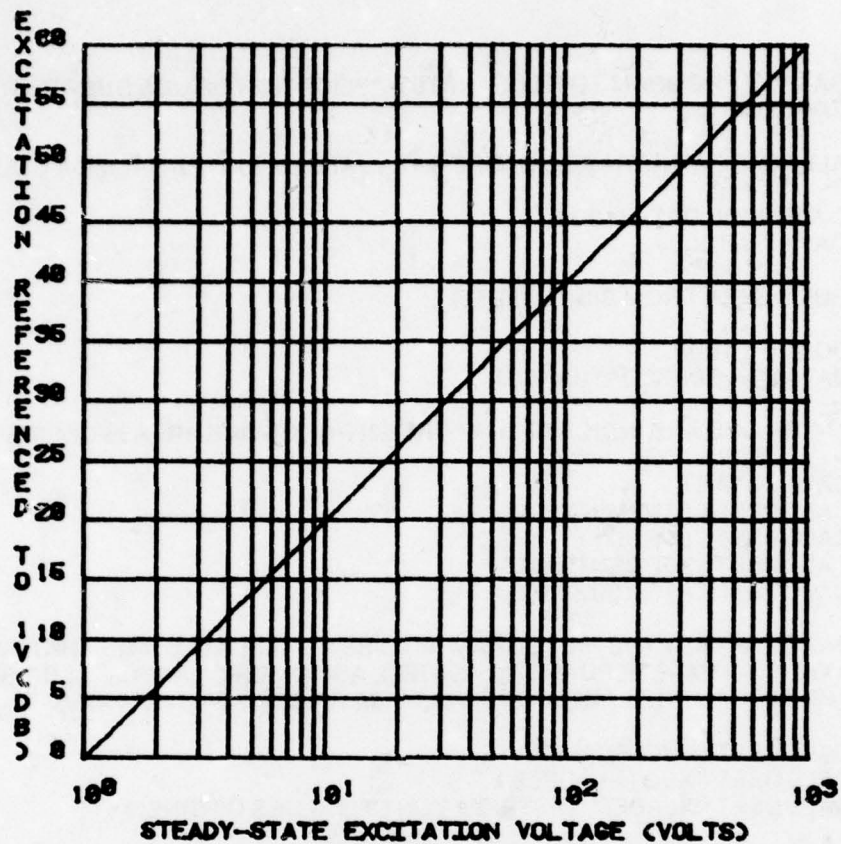
```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF SUBROUTINE
C   'UPLOT1'. DATA VALUES CONTAINED IN ARRAYS 'E2' AND 'DB' ARE PLOTTED
C   IN A SEMI-LOGARITHMIC COORDINATE SYSTEM. A GRIDDED AXES, AND
C   ALPHANUMERIC AXES LABELING ARE PROVIDED AS ADDITIONAL OPTIONS.
C   DEFAULT VALUE OF DEVICE AREA IS USED.
C
C   INITIALIZE ARRAYS, LOAD LABELS INTO LABEL ARRAYS.
C
C   DIMENSION DB(61),E2(61)
C   CHARACTER XLABEL*40,YLABEL*40
C   DATA XLABEL/'STEADY-STATE EXCITATION VOLTAGE (VOLTS)';/
C   DATA YLABEL/'EXCITATION REFERENCED TO 1V (DB)';/
C   DO 1 I=1,61
C     DB(I)=FLOAT(I-1)
C     E2(I)=10.0**(DB(I)/20.0)
1
C
C   ENTER GCS
C
C   CALL USTART
C
C   INDICATE THE DATA TO BE USED FOR AXES LABELING INFORMATION.
C
C   CALL UPSET ('XLABEL',XLABEL)
C   CALL UPSET ('YLABEL',YLABEL)
C   CALL UPSET ('TERMINATOR',';')
C
C   INDICATE THAT GRIDDED AXES, TOGETHER WITH NUMERIC AND ALPHA
C   LABELS ARE DESIRED. ALSO SPECIFY WHICH AXIS IS LINEAR, AND WHICH IS
C   LOGARITHMIC.
C
C   CALL USET ('GRIDAXES')
C   CALL USET ('XBOTHLABEL')
C   CALL USET ('YBOTHLABEL')
C   CALL USET ('LOGXAXIS')
C   CALL USET ('LINYAXIS')
C
C   CALL 'UPLOT1' TO PLOT A SINGLE CURVE OF 61 POINTS USING 'E2' AND 'DB'.
C   STANDARD LINE OPTION IS REQUESTED.
C
C   CALL UPLOT1 (E2,DB,61.0)
C   CALL UEND
C   STOP
C   END

```

### EXAMPLE VII-3



```

DIMENSION DB(61),E2(61)
CHARACTER XLABEL*48,YLABEL*48
DATA XLABEL/'STEADY-STATE EXCITATION VOLTAGE (VOLTS)',/
DATA YLABEL/'EXCITATION REFERENCED TO 1V(DB)',/
DO 1 I = 1, 61
  DB(I) = FLOAT(I-1)
  E2(I) = 18.8*(DB(I) / 20.8)
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UPSET ('XLABEL',XLABEL)
CALL UPSET ('YLABEL',YLABEL)
CALL USET ('GRIDAXES')
CALL USET ('XBOTHLABEL')
CALL USET ('YBOTHLABEL')
CALL USET ('XLOGAXIS')
CALL USET ('LINYAXIS')
CALL UPLT1 (E2,DB,61.8)
CALL UEND
STOP
END

```

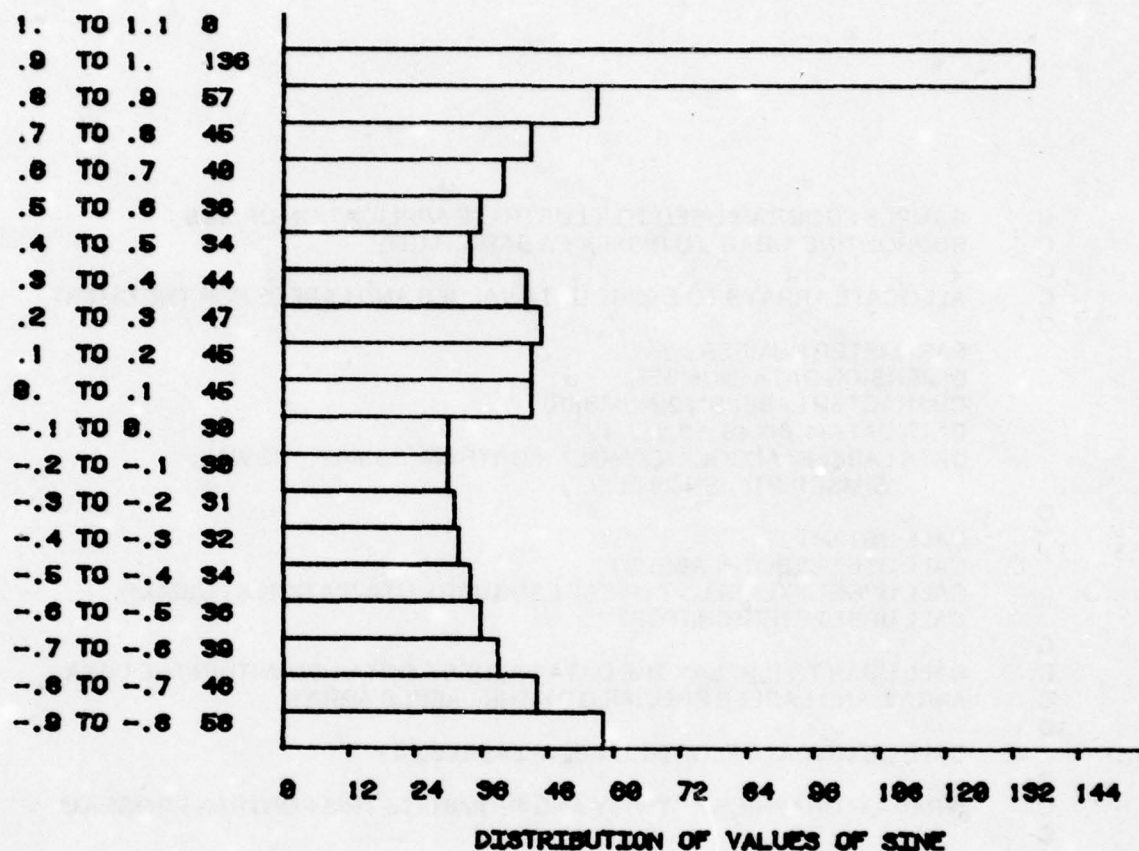


```

C      SAMPLE PROGRAM TO ILLUSTRATE APPLICATION OF GCS SUBROUTINE
C      'UHISTO'.
C
C      ALLOCATE AN ARRAY TO STORE DATA VALUES, AND INITIALIZE A COUNTER.
C
C      DIMENSION DATA (1000)
C      DATA XN/1000./
C
C      BUILD DATA FROM SINE VALUES
C
C      DO 2 I=1,1000
2 DATA(I) = SIN(FLOAT(I)/150.)
C
C      ENTER GCS AND INDICATE THAT THE ENTIRE DEVICE AREA IS DESIRED.
C
C      CALL USTART
C      CALL UPSET ('TERMINATOR',';')
C      CALL USET ('LARGE')
C      CALL USET ('PERCENTUNITS')
C      CALL UDAREA (0.,100.,0.,100.)
C
C      INDICATE THAT THE HISTOGRAM IS TO BE 'FULLSCALED' AND THAT THE X
C      AXIS IS TO HAVE ALPHABETIC AS WELL AS NUMERIC LABELS. ALSO SET
C      THE ALPHABETIC LABEL WHICH IS TO BE PRINTED ON THE X AXIS.
C
C      CALL USET ('FULLSCALE')
C      CALL USET ('XBOTHLABELS')
C      CALL USET ('XLABEL', 'DISTRIBUTION OF VALUES OF SINE;')
C
C      PROCESS THE DATA ARRAY USING 20 CELLS.
C
C      CALL UHISTO (DATA,XN,20.)
C
C      TERMINATION
C
C      CALL UEND
C      STOP
C      END

```

#### EXAMPLE VII-4



```

DIMENSION DATA(1800)
DATA XN/1800./
DO 2 I = 1, 1800
2 DATA(I) = SIN(FLOAT(I)/150.)
CALL USTART
CALL USET ('LARGE')
CALL USET ('PERCENTUNITS')
CALL UDAREA (8., 180., 8., 180.)
CALL USET ('FULLSCALE')
CALL USET ('XBOTHLABELS')
CALL UPSET ('TERMINATOR', ',')
CALL UPSET ('XLABEL', 'DISTRIBUTION OF VALUES OF SINE,')
CALL UHISTO (DATA, XN, 20.)
CALL UEND
STOP
END

```

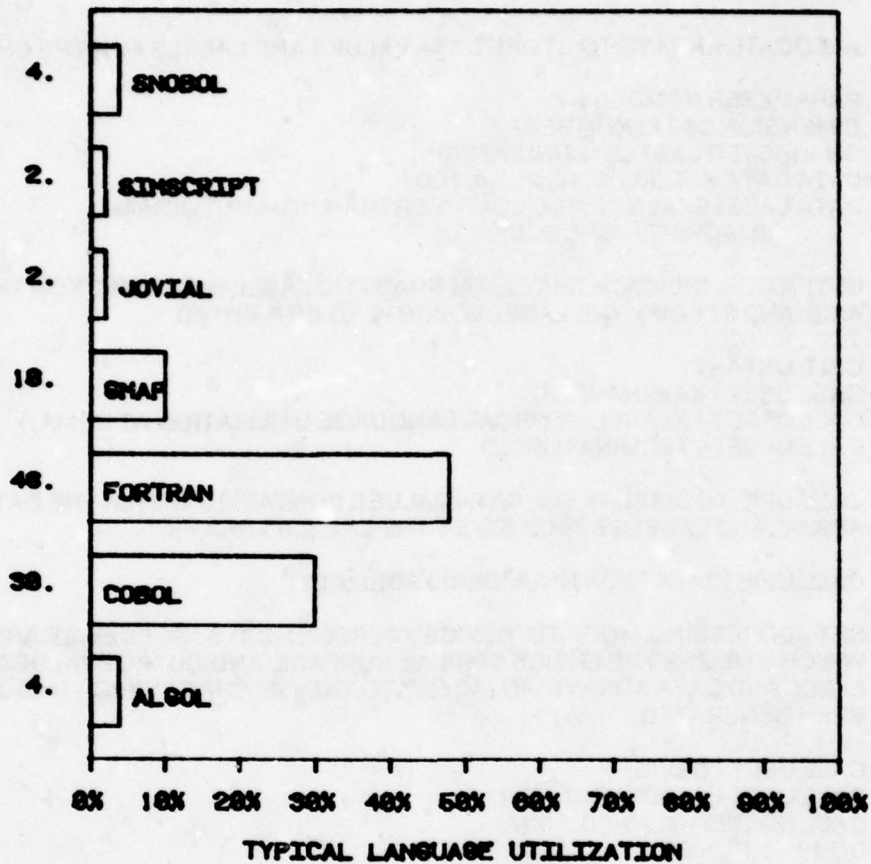
```

C      SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF GCS
C      SUBROUTINE 'UBAR' TO DISPLAY A BAR-CHART.
C
C      ALLOCATE ARRAYS TO STORE DATA VALUES AND LABELS FOR THE CHART.
C
C      PARAMETER NUMBER=7
C      DIMENSION DATA (NUMBER)
C      CHARACTER LABELS*12(NUMBER)
C      DATA DATA/4.,30.,48.,10.,2.,2.,4./
C      DATA LABELS/'ALGOL;', 'COBOL;', 'FORTRAN;', 'GMAP;', 'JOVIAL;',
C                   'SIMSCRIPT;', 'SNOBOL;'/
C
C      CALL USTART
C      CALL USET ('XBOTHLABELS')
C      CALL UPSET ('XLABEL', 'TYPICAL LANGUAGE UTILIZATION AT USMA:')
C      CALL UPSET ('TERMINATOR', ';')
C
C      CALL UBAR TO DISPLAY THE DATA VALUES CONTAINED WITHIN THE DATA
C      ARRAY, AND LABELS SPECIFIED BY THE LABELS ARRAY.
C
C      CALL UBAR (DATA,FLOAT(NUMBER),LABELS,12.)
C
C      WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THIS FORTRAN PROGRAM.
C
C      CALL UEND
C      STOP
C      END

```

#### EXAMPLE VII-5





```

PARAMETER NUMBER=7
DIMENSION DATA(NUMBER)
CHARACTER LABELS*12(NUMBER)
DATA DATA/4.,38.,48.,18.,2.,2.,4./
DATA LABELS/'ALGOL','COBOL','FORTRAN','SHAP','JOVIAL',
&          'SINSRIPT','SNOBOL','
CALL USTART
CALL UPSET ('TERMINATOR','')
CALL USET ('XBOTH LABELS')
CALL UPSET ('X LABEL','TYPICAL LANGUAGE UTILIZATION,')
CALL UBAR (DATA, FLOAT(NUMBER), LABELS, 12.)
CALL UEND
STOP
END

```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF GCS
C   SUBROUTINE 'UPIE'. IN ADDITION TO USING UPIE TO DISPLAY A PIE-CHART,
C   SOME ADDITIONAL GRAPHIC OUTPUT IS PERFORMED THROUGH USE OF
C   UPRNT1.
C
C   ALLOCATE ARRAYS TO STORE DATA VALUES AND LABELS FOR THE CHART.
C
C   PARAMETER NUMBER=7
C   DIMENSION DATA (NUMBER)
C   CHARACTER LABELS*12(NUMBER)
C   DATA DATA,Y/4.,30.,48.,10.,2.,2.,4.,100./
C   DATA LABELS/'ALGOL;', 'COBOL;', 'FORTRAN;', 'GMAP;', 'JOVIAL;',
C   'SIMSCRIPT;', 'SNOBOL;'/
C
C   ENTER GCS, INDICATE THAT AN ALPHABETIC LABEL IS DESIRED FOR THE X
C   AXIS, AND SPECIFY THE LABEL WHICH IS TO BE PRINTED.
C
C   CALL USTART
C   CALL USET ('XALPHABETIC')
C   CALL UPSET ('XLABEL', 'TYPICAL LANGUAGE UTILIZATION AT USMA;')
C   CALL UPSET ('TERMINATOR', ';')
C
C   CALL UPIE TO DISPLAY THE DATA VALUES CONTAINED WITHIN THE DATA
C   ARRAY, AND LABELS SPECIFIED BY THE LABELS ARRAY.
C
C   CALL UPIE (DATA,FLOAT(NUMBER),LABELS,12.)
C
C   SET ADDRESSING MODE TO 'DEVICE'/'PERCENTUNITS'. SET DEVICE AREA
C   WHICH UTILIZES THE ENTIRE DISPLAY SURFACE, AND OUTPUT VALUES IN
C   LABEL AND DATA ARRAYS, ADJACENT TO THE PIE-CHART WHICH HAS JUST
C   BEEN GENERATED.
C
C   CALL USET ('DEVICE')
C   CALL USET ('PERCENTUNITS')
C   CALL UDAREA (0.,100.,0.,100.)
C   DO 1 I=1,NUMBER
C   Y=Y-(100./FLOAT(NUMBER+1))
C   CALL UMOVE (0.,Y)
C   CALL UPRNT1 (LABELS(I),'TEXT')
C   CALL URPNT1 ('-',',', 'TEXT')
C   CALL UPRNT1 (DATA(I),'INTEGER')
C   CALL UPRNT1 (':', 'TEXT')
C 1  CONTINUE
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THIS FORTRAN PROGRAM.
C
C   CALL UEND
C   STOP
C   END

```

#### EXAMPLE VII-6

ALGOL - 4%

COBOL - 38%

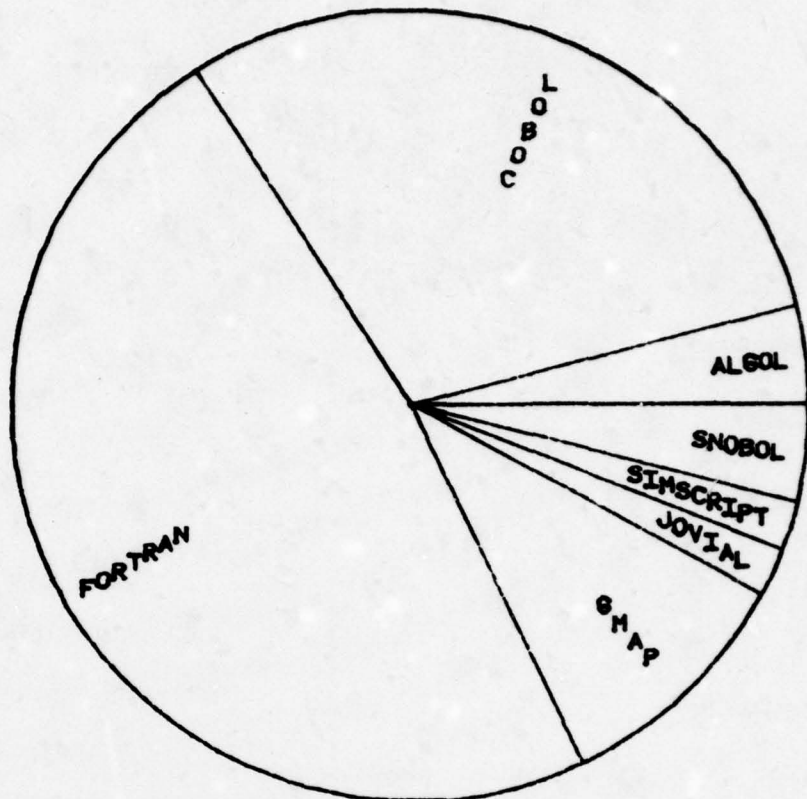
FORTRAN - 48%

SNAP - 18%

JOVIAL - 2%

SINSCRIPT - 2%

SNOBOL - 4%



TYPICAL LANGUAGE UTILIZATION

```
PARAMETER NUMBER=7
DIMENSION DATA(NUMBER)
CHARACTER LABELS(12)(NUMBER)
DATA DATA,Y/4.,38.,48.,18.,2.,2.,4.,188./
DATA LABELS/'ALGOL','COBOL','FORTRAN','SNAP','JOVIAL',
&          'SINSCRIPT','SNOBOL','
CALL USTART
CALL UPSET ('TERMINATOR','')
CALL USET ('XALPHABETIC')
CALL UPSET ('XLABEL','TYPICAL LANGUAGE UTILIZATION,')
CALL UPIE (DATA,FLOAT(NUMBER),LABELS,12.)
CALL USET ('DEVICE')
CALL USET ('PERCENTUNITS')
CALL UDAREA (8.,188.,8.,188.)
DO I = 1, NUMBER
Y = Y - (188. / FLOAT(NUMBER+1))
CALL UMOVE (8.,Y)
CALL UPRT1 (LABELS(I),'TEXT')
CALL UPRT1 (' - ','TEXT')
CALL UPRT1 (DATA(I),'INTEGER')
CALL UPRT1 ('X','TEXT')
CONTINUE
CALL UEND
STOP
END
```



## CHAPTER VIII

### COORDINATE SYSTEMS AND TRANSFORMATIONS

#### **An Overview of GCS Coordinate Systems**

The GCS coordinate system options have been designed to provide adequate flexibility for the user in order to enable him to work in whatever type of coordinate system environment is most natural and convenient for him to use at a particular point in his program. There are several areas in which basic options exist; however, it is imperative that the user fully understand the basic coordinate system under GCS before attempting to alter any of the default options.

The GCS default coordinate system which is defined at the start of a GCS program (via a CALL USTART) is a simple, pre-defined Cartesian (rectangular) X-Y coordinate system. No special options such as semi-logarithmic or log-log plotting are in effect. Axes are not rotated with respect to the screen or plotted and no special scaling options are in force. Distances are measured from the origin, not incrementally from the last point plotted. As described in Chapter III, any drawing done in this coordinate system may be viewed through an adjustable size window which can automatically clip off irrelevant or distracting parts of the picture.

A GCS programmer may specify that he wishes to use one of three coordinate systems when working in two dimensions:

- Cartesian (rectangular)
- Log-log or semi-logarithmic rectangular
- Polar coordinates.

Points or lines to be plotted may be identified either by absolute coordinate position or relative to the last plotted point (incremental). The scale at which plotting is to occur may be user-specified in virtual space which can be projected onto the plotting device at any desired scale or it may be directly controlled in terms of dimensions of the plotting device. These dimensions may be specified in inches, centimeters, font-units (alphanumeric character heights and widths) percent-units, or raster units.

In addition to the basic or SYSTEM axis, the user may define any number of secondary USER axis systems which may be rotated, translated, and scaled differently from the SYSTEM axis. Such secondary USER axes may be defined relative to the basic SYSTEM axis or relative to a previously defined USER axis. This feature facilitates the definition of complex coordinate systems where, for example, one moves freely from coordinates based upon a location on a satellite circling the earth. Similarly, the use of simultaneously maintaining a secondary axis whose origin is a point on the curve, oriented controlled independently of one another and defined either with respect to the SYSTEM axis or cumulatively based on successive transformations on a series of different USER axes. Since the rotations and scaling in GCS are accomplished by standard matrix operations one can obtain mathematically accurate visual representations of highly abstract vector spaces such as those common to many linear programming applications.

#### **Coordinate System Definition**

In order to provide the capabilities discussed above, GCS has incorporated a very powerful secondary axis system which allows the user to define a new axis origin. The new origin can be displaced from the current origin, oriented about the new center point, and/or have a different unit length for the X and for the Y axis. The new axis is defined through subroutine UCOSYS which is called by the following sequence:

#### CALL UCOSYS (DX,DY,SCLX,SCLY,ANGLE)

DX and DY are used to specify (in current units of the current coordinate system) the position of the origin for the new axis. SCLX and SCLY define the scale of the new axis unit lengths with respect to their sizes in the current axis system. ANGLE is used to specify the rotation (in current angular units) of the new axis about its origin.

The key rule to be remembered is that every axis system has an origin which is coordinate location (0,0); all rotation occurs about this center point. The SYSTEM origin is the primary origin for the default axis system. In virtual space, the SYSTEM origin is at virtual location (0,0); this location may or may not be within the window defined by the user. In device space, it is a point on the display surface (usually at the lower left corner) which is at raster position (0,0). At any time, the user can insure that his coordinate addressing is interpreted with respect to the primary origin by making the following call to USET prior to addressing:

#### CALL USET ('SYSTEMAXIS')

To request that coordinate specifications be interpreted with respect to the user's current axis system, the following option is used:

#### CALL USET ('USERAXIS')

The default user axis coincides with the system axis. When a new user axis is created by subroutine UCOSYS, the GCS status is automatically switched to 'USERAXIS'. When UCOSYS is invoked, there are actually two USER axes created. They are either identical to each other, or one 'lags' (in a mathematical sense) behind the other by one USER axis definition. One of these USER axes is called the WORKING axis and the other is called the REFERENCE axis. The REFERENCE axis is identical to, or 'behind' the WORKING axis. Under default conditions, the REFERENCE and WORKING axes are identical to each other; furthermore, these axes initially coincide with the SYSTEM axis. When a new USER axis is defined, it is created from the REFERENCE axis and entered as the WORKING axis transform. If UCOSYS is in the cumulative mode, then the new WORKING axis definition is placed back into the REFERENCE axis transform. Thus, the REFERENCE axis may be viewed as a permanent axis, while the WORKING axis may be considered as only temporary in nature.

When multiple coordinate systems are specified by the user, the new axis is computed independently of the previous USER axis. This is the new axis which is computed from the last permanent axis (usually the SYSTEM axis) and replaces any previously defined USER axis. This feature, which is the default condition under GCS, is selected by the following call:

#### CALL USET ('WORKINGAXIS')

For advanced problems, the user may wish to define multiple coordinate systems that are computed on a cumulative basis. That is, the new axis is computed from the last permanent axis, and becomes the new permanent axis. This option may be specified through the following call:

#### CALL USET ('REFERENCEAXIS')

There exist several useful subroutines within GCS which are subsets of subroutine UCOSYS, and facilitate the definition of a secondary axis at the current beam position. Subroutine UROTAT creates a new axis rotated about an origin defined by the current beam position. Subroutine USCALE defines an axis at the current beam position having a specified X axis and Y axis scaling. Subroutine UORIGIN defines a new axis at the current beam position with no change of scale or orientation. These subroutines may be invoked by the calling sequences:

CALL UROTAT (ANGLE)  
CALL USCALE (SCLX,SCLY)  
CALL UORIGN

ANGLE, SCLX, and SCLY are interpreted in the same manner described for subroutine UCOSYS. After the subroutines are invoked, the current beam position is associated with coordinate location (0,0).

In certain situations, such as the display of unrelated objects, multiple independent secondary axis transformations must be created simultaneously. In order to operate upon and modify one system, it is necessary to save the previous system and later restore it as needed. This facility is available under GCS through the use of the following subroutines:

CALL USVTR (ARRAY)

which is used to save the current transform, and

CALL UNSVTR (ARRAY)

which is called in order to restore a transformation from the array ARRAY, where ARRAY is a 21 word dimensioned variable. Note that a call to USVTR does not affect the status of the Graphics Compatibility System, but that a call to UNSVTR causes the restoration of all variables and switches to the values that they held when the companion USVTR was invoked. The user should not attempt to invoke UNSVTR for an array without previously saving a transform into that array by use of USVTR.

The secondary axis transformation feature of GCS is a very powerful mathematical system which is quite useful to the beginning as well as advanced graphics programmer. However, certain combinations of axis composition and specification will produce results which are mathematically correct but not easily understood. For example, the composition of a cumulative axis from an axis with non-uniform X and Y scaling will introduce a skew factor. For additional information on the secondary axis implementation under GCS, interested users are directed to the subroutine description section.

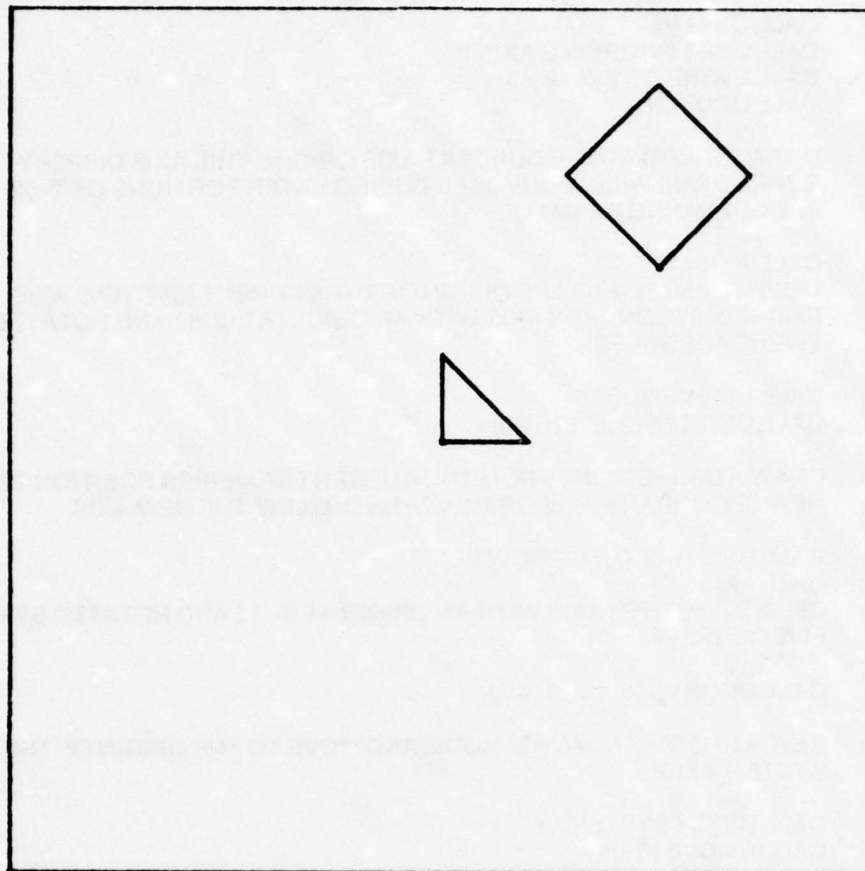


```

C      SAMPLE PROGRAM USED TO ILLUSTRATE THE USE OF GCS SUBROUTINE
C      'UCOSYS'. TWO DISPLAYS WILL BE DEFINED: A TRIANGLE WILL BE PLOTTED
C      IN THE DEFAULT COORDINATE SYSTEM; AND A SQUARE WILL BE DRAWN IN
C      A USER-DEFINED SECONDARY AXIS SYSTEM.
C
C      ENTER GCS, DEFINE A NEW VIRTUAL WINDOW, AND OUTLINE WINDOW.
C
C      CALL USTART
C      CALL USET ('WORKINGAXIS')
C      CALL UWINDO (-10.,10.,-10.,0.)
C      CALL UOUTLN
C
C      PROVIDE PEN COMMANDS TO DRAW A TRIANGLE IN THE DEFAULT GCS
C      COORDINATE SYSTEM.
C
C      CALL UMOVE (0.,0.)
C      CALL UPEN (2.,0.)
C      CALL UPEN (0.,2.)
C      CALL UPEN (0.,0.)
C
C      DEFINE A SECONDARY AXIS WITH AN ORIGIN AT (5.,4.), HAVING THE SAME X
C      AND Y SCALE FACTORS AS THE DEFAULT SYSTEM, AND ROTATED BY 45
C      DEGREES.
C
C      CALL UCOSYS (5.,4.,1.,1.,45.)
C
C      PROVIDE PEN COMMANDS TO DRAW A SQUARE WITHIN THE SECONDARY
C      AXIS THAT WE HAVE JUST DEFINED.
C
C      CALL UMOVE (0.,0.)
C      CALL URECT (3.,3.)
C
C      WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THE FORTRAN PROGRAM.
C
C      CALL UEND
C      STOP
C      END

```

011  
EXAMPLE VIII-1



```

CALL USTART
CALL USETC('WORKINGAXIS')
CALL UGENDO (-18.,18.,-18.,18.)
CALL UOUTLN
CALL UMOVE (8.,8.)
CALL UPEN (2.,8.)
CALL UPEN (8.,2.)
CALL UPEN (8.,8.)
CALL UCOSYS (5.,4.,1.,1.,45.)
CALL UMOVE (8.,8.)
CALL URECT (3.,3.)
CALL UEND
STOP
END

```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE COMPOSITION OF MULTIPLE
C   USER AXES. THE AXIS DEFINITION WILL BE PERFORMED IN THE 'NON-
C   CUMULATIVE' OR 'WORKING AXIS' MODE. ENTER GCS, DEFINE A NEW
C   VIRTUAL WINDOW, AND OUTLINE WINDOW.
C
C   CALL USTART
C   CALL USET ('WORKING AXIS')
C   CALL UWINDO (-2.,8.,-2.,8.)
C   CALL UCUTLN
C
C   DRAW AN AXIS AT THE CURRENT AXIS ORIGIN. THIS AXIS DISPLAY
C   SUBROUTINE WILL BE INVOKED DURING LATER PORTIONS OF THIS
C   PROGRAMMING EXAMPLE.
C
C   CALL AXIS
C   MOVE BEAM/PEN TO THE ORIGIN OF THE CURRENT (SYSTEM) AXIS, THEN
C   DEFINE A SECONDARY AXIS WITH AN ORIGIN AT (2.,5.) AND ROTATED BY
C   TWENTY DEGREES.
C
C   CALL UMOVE (0.,0.)
C   CALL UCOSYS (2.,5.,1.,1.,20)
C
C   DRAW A DASHED LINE FROM THE CURRENT BEAM/PEN POSITION TO THE
C   NEW SECONDARY AXIS ORIGIN, THEN OUTLINE THE NEW AXIS.
C
C   CALL UPEN1 (0.,0.,'DARROW')
C   CALL AXIS
C   DEFINE ANOTHER AXIS WITH AN ORIGIN AT (5.,1.) AND ROTATED BY FORTY-
C   FIVE DEGREES.
C
C   CALL UCOSYS (5.,1.,1.,1.,45.)
C
C   REVERT TO 'SYSTEMAXIS' MODE, AND MOVE TO THE ORIGIN OF THE
C   SYSTEM AXIS.
C
C   CALL USET ('SYSTEMAXIS')
C   CALL UMOVE (0.,0.)
C
C   RETURN TO 'USERAXIS' MODE, AND DRAW A DASHED LINE FROM THE ORIGIN
C   OF THE SYSTEM AXIS TO THE ORIGIN OF THE USER AXIS. AFTER THIS IS
C   DONE, OUTLINE THE CURRENT USER AXIS.
C
C   CALL USET ('SYSTEMAXIS')
C   CALL UMOVE (0.,0.)
C   RETURN TO 'USERAXIS' MODE, AND DRAW A DASHED LINE FROM THE ORIGIN
C   OF THE SYSTEM AXIS TO THE ORIGIN OF THE USER AXIS. AFTER THIS IS
C   DONE, OUTLINE THE CURRENT USER AXIS.
C
C   CALL USET ('USERAXIS')
C   CALL UPEN1 (0.,0.,'DARROW')
C   CALL AXIS
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE A FORTRAN PROGRAM.
C
C   CALL UEND
C   STOP
C   END

```

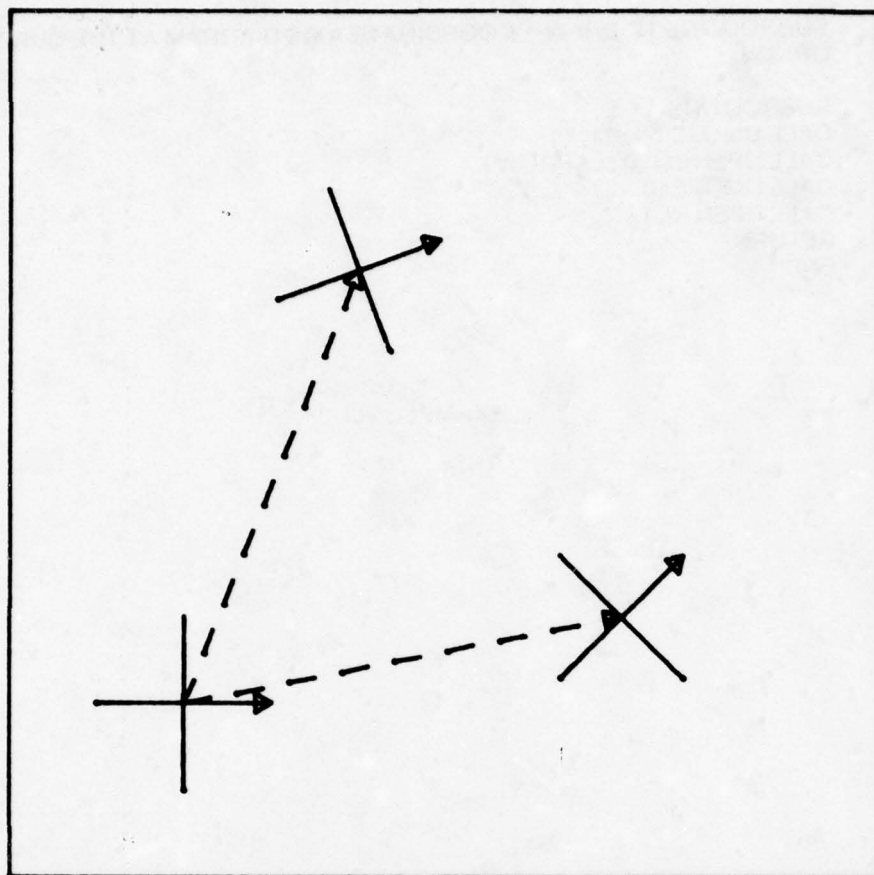


C  
C  
C  
C

SUBROUTINE TO DRAW A COORDINATE AXIS DIAGRAM AT THE CURRENT ORIGIN.

SUBROUTINE AXIS  
CALL UMOVE (-1.,0.)  
CALL UPEN1 (1.,0., 'L'ARROW')  
CALL UMOVE (0.,-1.)  
CALL UPEN (0.,1.)  
RETURN  
END

EXAMPLE VIII-2



```

CALL USTART
CALL USET ('WORKINGAXIS')
CALL UNDO (-2.,8.,-2.,8.)
CALL UOUTLN
CALL AXIS
CALL UNOVE (8.,8.)
CALL UCOSYS (2.,5.,1.,1.,28.)
CALL UPEN1 (8.,8.,'DARROW')
CALL AXIS
CALL UCOSYS (5.,1.,1.,1.,45.)
CALL USET ('SYSTEMAXIS')
CALL UNOVE (8.,8.)
CALL USET ('USERAXIS')
CALL UPEN1 (8.,8.,'DARROW')
CALL AXIS
CALL UNDO
STOP
END
SUBROUTINE AXIS
CALL UNOVE (-1.,8.)
CALL UPEN1 (1.,8.,'LARROW')
CALL UNOVE (8.,-1.)
CALL UPEN (8.,1.)
RETURN
END

```

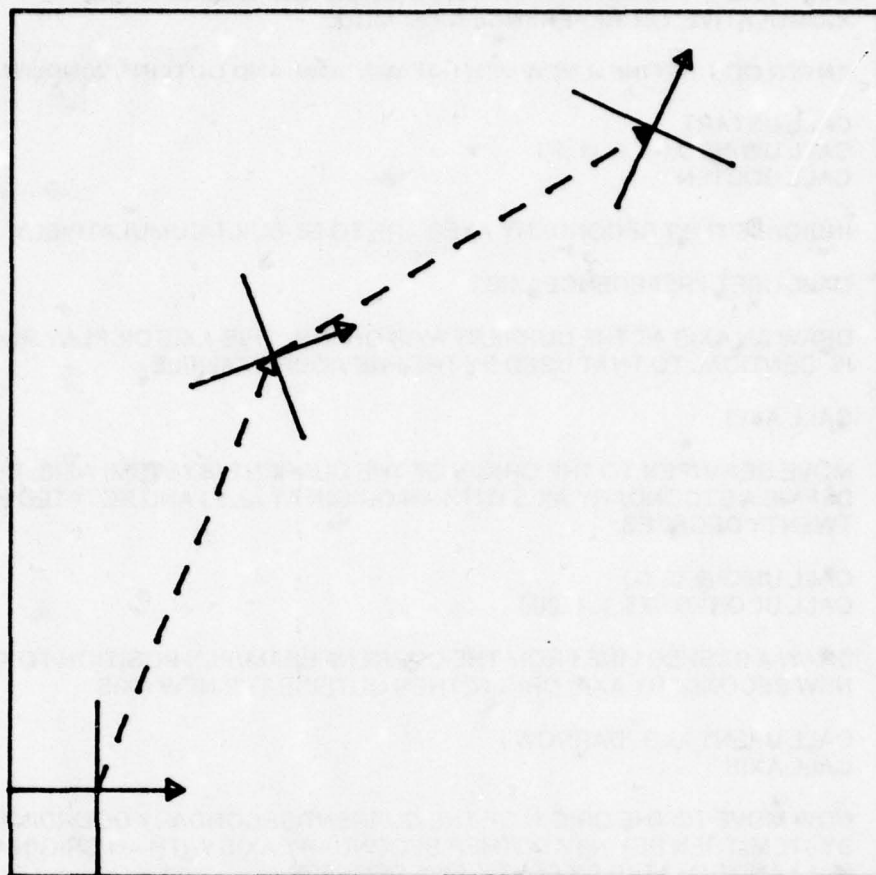
```

C   EXAMPLE PROGRAM USED TO ILLUSTRATE COMPOSITION OF MULTIPLE
C   USER AXES. THE AXIS DEFINITION WILL BE PERFORMED IN THE
C   'CUMULATIVE' OR 'REFERENCE AXIS' MODE.
C
C   ENTER GCS, DEFINE A NEW VIRTUAL WINDOW, AND OUTLINE WINDOW.
C
C   CALL USTART
C   CALL UWINDO (-1.,9.,-1.,9.)
C   CALL UOUTLN
C
C   INDICATE THAT SECONDARY AXES ARE TO BE BUILT CUMULATIVELY.
C
C   CALL USET ('REFERENCEAXIS')
C
C   DRAW AN AXIS AT THE CURRENT AXIS ORIGIN. THIS AXIS DISPLAY ROUTINE
C   IS IDENTICAL TO THAT USED BY THE PREVIOUS EXAMPLE.
C
C   CALL AXIS
C
C   MOVE BEAM/PEN TO THE ORIGIN OF THE CURRENT (SYSTEM) AXIS, THEN
C   DEFINE A SECONDARY AXIS WITH AN ORIGIN AT (2.,5.) AND ROTATED BY
C   TWENTY DEGREES.
C
C   CALL UMOVE (0.,0.)
C   CALL UCOSYS (2.5.,1.,1.,20.)
C
C   DRAW A DASHED LINE FROM THE CURRENT BEAM/PEN POSITION TO THE
C   NEW SECONDARY AXIS ORIGIN, THEN OUTLINE THE NEW AXIS.
C
C   CALL UPEN1 (0.,0.,'DARROW')
C   CALL AXIS
C
C   NOW MOVE TO THE ORIGIN OF THE CURRENT SECONDARY COORDINATE
C   SYSTEM, THEN DEFINE ANOTHER SECONDARY AXIS WITH AN ORIGIN AT
C   (5.,1.) AND ROTATED BY FORTY-FIVE DEGREES.
C
C   CALL UMOVE (0.,0.)
C   CALL UCOSYS (5.,1.,1.,45.)
C
C   DRAW A DASHED LINE FROM THE CURRENT BEAM/PEN POSITION (THE
C   ORIGIN OF THE PREVIOUS SECONDARY AXIS) TO THE ORIGIN OF A CURRENT
C   SECONDARY AXIS, THEN OUTLINE THE CURRENT AXIS.
C
C   CALL UPEN1 (0.,0.,'DARROW')
C   CALL AXIS
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE A FORTRAN PROGRAM.
C
C   CALL UEND
C   STOP
C   END
C
C   SUBROUTINE AXIS
C   CALL UMOVE (-1.,0.)
C   CALL UPEN1 (1.,0.,'LARROW')
C   CALL UMOVE (0.,-1.)
C   CALL UPEN1 (0.,1.)
C   RETURN
C   END

```

EXAMPLE VIII-3





```

CALL USTART
CALL UUNDO (-1.,9.,-1.,9.)
CALL UOUTLN
CALL USET ('REFERENCEAXIS')
CALL AXIS
CALL UMOVE (8.,8.)
CALL UCOSYS (2.,5.,1.,1.,28.)
CALL UPEN1 (8.,8.,'DARROW')
CALL AXIS
CALL UMOVE (8.,8.)
CALL UCOSYS (5.,1.,1.,1.,45.)
CALL UPEN1 (8.,8.,'DARROW')
CALL AXIS
CALL UEND
STOP
END
SUBROUTINE AXIS
CALL UMOVE (-1.,8.)
CALL UPEN1 (1.,8.,'LARROW')
CALL UMOVE (8.,-1.)
CALL UPEN1 (8.,1.)
RETURN
END

```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF GCS
C   SUBROUTINE 'UCOSYS' TO DEFINE A DYNAMIC SECONDARY AXIS SYSTEM.
C   THE EXAMPLE CALCULATES THE HORIZONTAL AND VERTICAL
C   COMPONENTS OF VELOCITY AND DISPLACEMENT FOR A PROJECTILE
C   WHICH TUMBLES DURING FLIGHT. THIS EXAMPLE IS IDENTICAL TO THE
C   PREVIOUS PROGRAM WITH THE EXCEPTION OF THE ADDED ROTATIONAL
C   COMPONENT.
C
C   INITIALIZE TIME, ROTATIONAL ANGLE, VELOCITY, AND DISPLACEMENTS
C
C   DATA T,THETA,VO,XO,YO/0.,75.,107.,-180.,0./
C
C   ENTER GCS, DEFINE DISPLACEMENT WINDOW, AND OUTLINE THIS WINDOW.
C
C   CALL USTART
C   CALL USET ('WORKINGAXIS')
C   CALL UWINDO (-225.,225.,-15.,95.)
C   CALL UOUTLN
C
C   MOVE INITIAL DISPLACEMENT COORDINATES, SPECIFY THAT SOFTWARE
C   CHARACTERS ARE DESIRED, LINE CHARACTER SIZE, AND SET DASH SPEC.
C
C   CALL UMOVE (XO,YO)
C   CALL USET ('SOFTWARE')
C   CALL UPSET ('HORIZONTAL',6.)
C   CALL UPSET ('VERTICAL',3.)
C   CALL UPSET ('SETDASH',92.)
C
C   DEFINE LOOP TO CALCULATE HORIZONTAL AND VERTICAL
C   DISPLACEMENTS.
C
C   DO 1 T=1,11
C     X=(.707*VO*T)+XO
C     Y=T*(.707*VO-(16.*T))+YO
C
C   SPECIFY THE SECONDARY AXIS AT COMPUTED DISPLACEMENT
C   COORDINATES THEN DRAW THE FLIGHT PATH OF THE PROJECTILE TO THIS
C   ORIGIN.
C
C   CALL UCOSYS (X,Y,1.,1.,THETA)
C   CALL UPEN1 (0.,0., 'DASH')
C
C   SETUP A NEW WINDOW FOR DRAWING THE PROJECTILE ITSELF WITHIN THE
C   ROTATED COORDINATE SYSTEM WE HAVE JUST DEFINED. A DISTORTED 'D'
C   REPRESENTS THE PROJECTILE.
C
C   CALL UWINDO (0.,100.,0.,100.)
C   CALL UWHERE (X,Y)
C   CALL UPEN1 (X,Y,'DD')
C
C   REDEFINE DISPLACEMENT WINDOW, UPDATE TIME AND ROTATIONAL
C   ANGLE.
C
C   CALL UWINDO (-225.,225.,-15.,95.)
C   T=T+0.4758
C   THETA=THETA+57.
1  CONTINUE

```

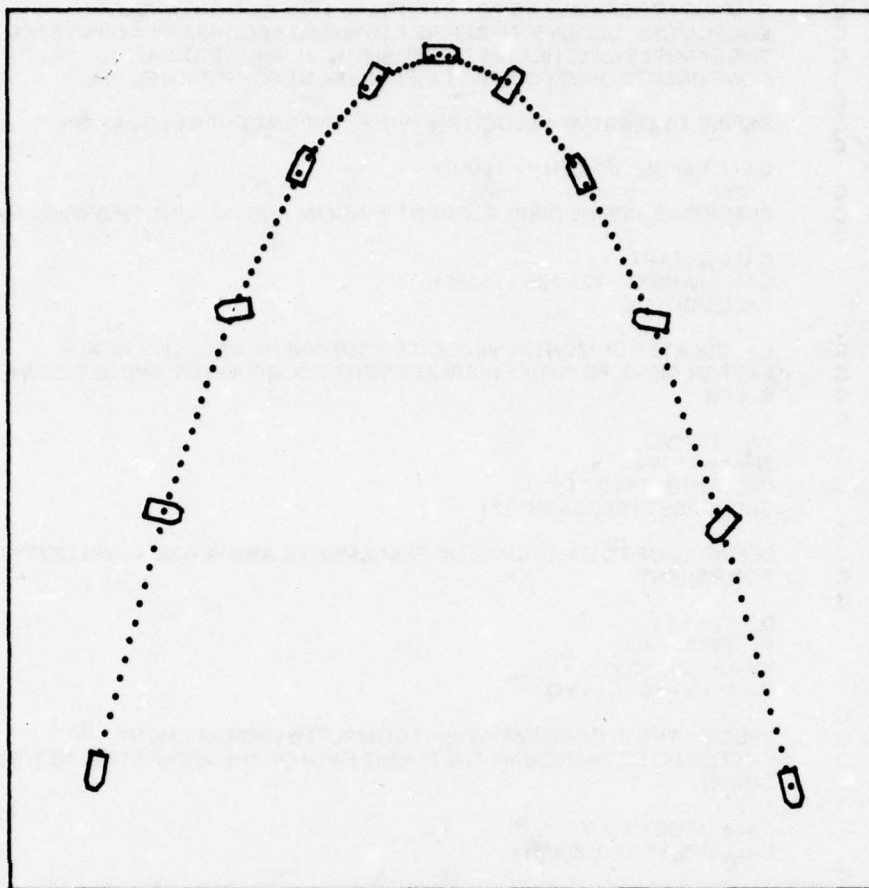
C  
C  
C

WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THIS FORTRAM PROGRAM.

CALL UEND  
STOP  
END

EXAMPLE VIII-4





```

DATA T,THETA,V8,X8,Y8/8.,75.,187.,-188.,8./
CALL USTART
CALL USET ('WORKINGAXIS')
CALL UWINDO (-225.,225.,-15.,85.)
CALL UOUTLN
CALL UMOVE (X8,Y8)
CALL USET ('SOFTWARE')
CALL UPSET ('HORIZONTAL',8.)
CALL UPSET ('VERTICAL',3.)
CALL UPSET ('SETDASH',82.)
DO I X = 1, 11
  X = (.787*V8*WT) + X8
  Y = T + (.787 * V8 - (16.*WT)) + Y8
  CALL UCOSYS (X,Y,1.,1.,THETA)
  CALL UPENI (8.,8.,'DASH')
  CALL UWINDO (8.,188.,8.,188.)
  CALL UWHERE (X,Y)
  CALL UPENI (X,Y,'DD')
  CALL UWINDO (-225.,225.,-15.,85.)
  T = T + 8.4758
  THETA = THETA + 57.
I CONTINUE
CALL UEND
STOP
END

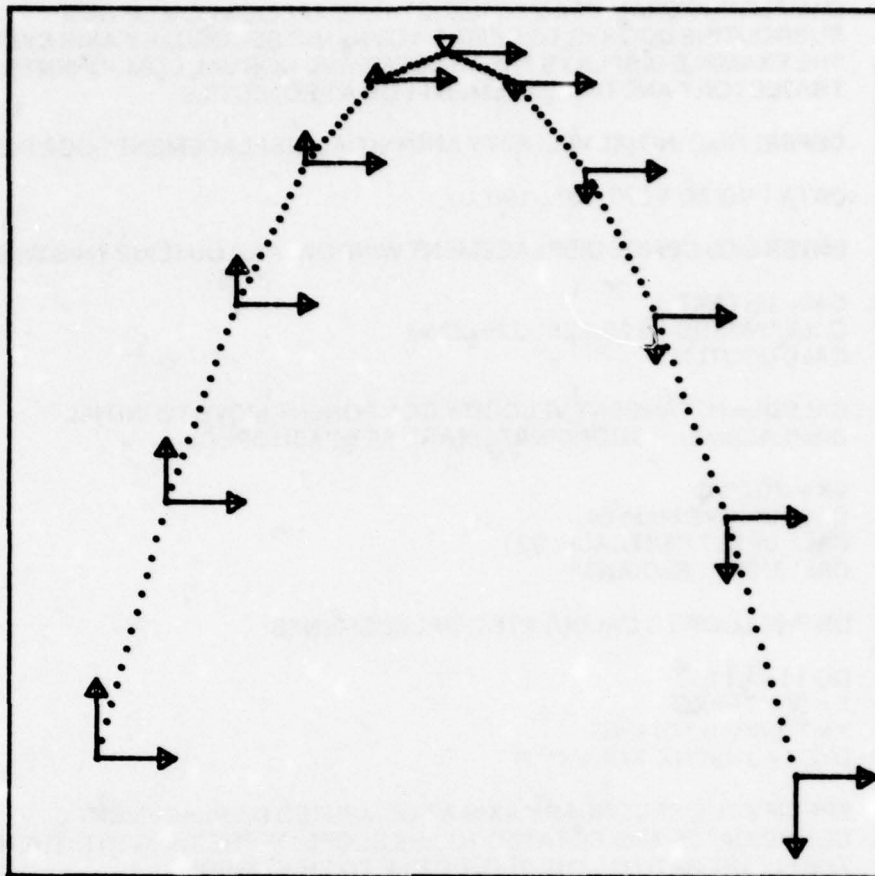
```

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF GCS
C   SUBROUTINE 'UCOSYS' TO DEFINE A DYNAMIC SECONDARY AXIS SYSTEM.
C   THE EXAMPLE CALCULATES THE HORIZONTAL AND VERTICAL
C   COMPONENTS OF VELOCITY & DISPLACEMENT FOR A PROJECTILE.
C
C   DEFINE TIME, INITIAL VELOCITY, & INITIAL DISPLACEMENT LOCATION.
C
C   DATA T,VO,XO,YO/0.,107.,-190.0./
C
C   ENTER GCS, DEFINE DISPLACEMENT WINDOW, AND OUTLINE THIS WINDOW.
C
C   CALL USTART
C   CALL UWINDO (-225,225.,-15.,95.)
C   CALL UOUTLN
C
C   CALCULATE HORIZONTAL VELOCITY COMPONENT, VELOCITY SCALE
C   FACTOR, MOVE TO INITIAL DISPLACEMENT COORDINATES, AND SET DASH
C   SPECS.
C
C   VS=.707*VO
C   ZETA=11.*VX
C   CALL UMOVE (XO,YO)
C   CALL UPSET ('SETDASH',92.)
C
C   DEFINE LOOP TO CALCULATE DISPLACEMENTS, AND VERTICAL VELOCITY
C   COMPONENT.
C
C   DO 1 I=1,11
C   X=(FX*T)+XO
C   VY=2.*(VX-32.*T)
C   Y=T*(VX-(16.*T))+YO
C
C   SPECIFY THE SECONDARY AXIS AT COMPUTED DISPLACEMENT
C   COORDINATES THEN DRAW THE FLIGHT PATH OF THE PROJECTILE TO THIS
C   ORIGIN.
C
C   CALL UCOSYS (X,Y,1.,1.,0.)
C   CALL UPEN1 (0.,0., 'DASH')
C
C   DEFINE A NEW WINDOW FOR THE VELOCITY AND PLOT THE HORIZONTAL &
C   VERTICAL COMPONENTS OF THE PROJECTILE'S VELOCITY.
C
C   CALL UWINDO (0.,ZETA,-ZETA,ZETA)
C   CALL USET ('RELATIVE')
C   CALL UPEN1 (VX,0., 'LARROW')
C   CALL UMOVE (-VX,0.)
C   CALL UPEN1 (0.,VY, 'LARROW')
C   CALL UMOVE (0.,-VY)
C
C   REDEFINE DISPLACEMENT WINDOW, UPDATE TIME, AND CONTINUE A LOOP.
C
C   CALL USET ('ABSOLUTE')
C   CALL UWINDO (-225,225.,-15.,95.)
C   T=T+4758
C 1  CONTINUE
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THIS FORTRAN PROGRAM.
C
C   CALL UEND
C   STOP
C   END

```

EXAMPLE VIII-5



```

DATA T,V8,X8,Y8/8.,187.,-188.,8./
CALL USTART
CALL USET ('WORKINGAXIS')
CALL UNINDO (-225.,225.,-15.,85.)
CALL UOUTLN
VX = .787 * V8
ZETA = 11. * VX
CALL UMOVE (X8,Y8)
CALL UPSET ('SETDASH',92.)
DO I I = 1, 11
  X = (VX*IT) + X8
  VY = 2. * (VX - (92.*IT))
  Y = T * (VX - (10.*IT)) + Y8
  CALL UCOSYS (X,Y,1.,1.,8.)
  CALL UPEN1 (8.,8.,'DASH')
  CALL UNINDO (8.,ZETA,-ZETA,ZETA)
  CALL USET ('RELATIVE')
  CALL UPEN1 (VX,8.,'LARRON')
  CALL UMOVE (-VX,8.)
  CALL UPEN1 (8.,VY,'LARRON')
  CALL UMOVE (8.,-VY)
  CALL USET ('ABSOLUTE')
  CALL UNINDO (-225.,225.,-15.,85.)
  T = T + .4768
I CONTINUE
CALL UEND
STOP
END

```

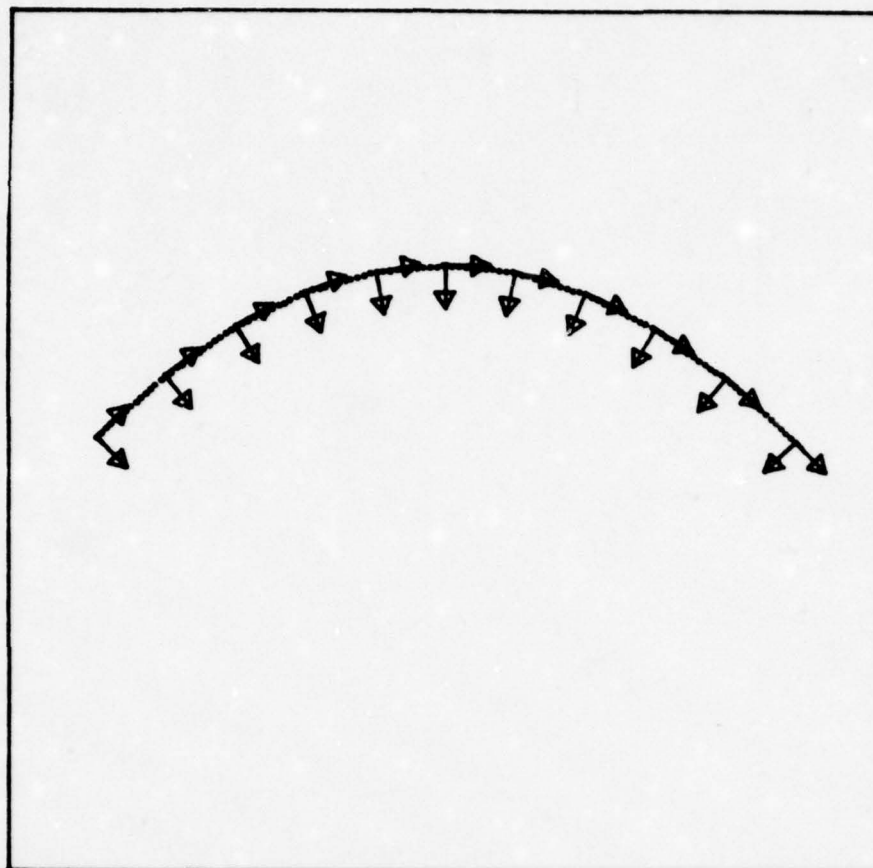


```

C   SAMPLE PROGRAM USED TO ILLUSTRATE APPLICATION OF GCS
C   SUBROUTINE UCOSYS TO DEFINE A DYNAMIC SECONDARY AXIS SYSTEM.
C   THE EXAMPLE DISPLAYS THE TANGENT AND NORMAL COMPONENTS OF
C   TRAJECTORY AND DISPLACEMENT FOR A PROJECTILE.
C
C   DEFINE TIME INITIAL VELOCITY AND INITIAL DISPLACEMENT LOCATION.
C
C   DATA T,VO,XO,YO/0.,107.,-190.,0./
C
C   ENTER GCS DEFINE DISPLACEMENT WINDOW, AND OUTLINE THIS WINDOW.
C
C   CALL USTART
C   CALL UWINDO (-225.,225.,-225.,225.)
C   CALL UOUTLN
C
C   CALCULATE TANGENT VELOCITY COMPONENT MOVE TO INITIAL
C   DISPLACEMENT COORDINATES, AND SET DASH SPECS.
C
C   VX=.707*VO
C   CALL UMOVE (XO,YO)
C   CALL UPSET ('SETDASH',92.)
C   CALL USET ('RADIANS')
C
C   DEFINE LOOP TO CALCULATE DISPLACEMENTS
C
C   DO 1 I=1,11
C   X=(VX*T)+XO
C   Y=T*(VX-16.*T))+YO
C   DYDX=1-(32*(X-XO)/VX**2)
C
C   SPECIFY THE SECONDARY AXIS AT COMPUTED DISPLACEMENT
C   COORDINATES AND ROTATED TO THE SLOPE OF THE TANGENT. THEN DRAW
C   THE FLIGHT PATH OF THE PROJECTILE TO THIS ORIGIN.
C
C   CALL UCOSYS (X,Y,1.,1.,ATAN(DYDX))
C   CALL UPEN1 (0.,0.,'DASH')
C
C   DEFINE A NEW WINDOW AND PLOT THE TANGENT & NORMAL COMPONENTS
C   OF THE PROJECTILE'S TRAJECTORY.
C
C   CALL UWINDO (0.,1.,-1.,0.)
C   CALL USET ('RELATIVE')
C   CALL UPEN1 (.05,0.,'LARROW')
C   CALL UMOVE (-.05,0.)
C   CALL UPEN1 (0.,-.05,'LARROW')
C   CALL UMOVE (0.,.05)
C
C   REDEFINE DISPLACEMENT WINDOW, UPDATE TIME, AND CONTINUE A LOOP.
C
C   CALL USET ('ABSOLUTE')
C   CALL UWINDO (-225.,225.,-225.,225.)
C   T=1+4/58
C   1 CONTINUE
C
C   WRAP-UP GRAPHICS ACTIVITY AND TERMINATE THIS FORTRAN PROGRAM.
C
C   CALL UEND
C   STOP
C   END

```

EXAMPLE IX-6



```

DATA T,V8,X8,Y8/8.,187.,-180.,8./
CALL USTART
CALL USET ('WORKINGAXIS')
CALL UNINDO (-225.,225.,-225.,225.)
CALL UOUTLN
VX = .787 * V8
CALL UMOVE (X8,Y8)
CALL UPSET ('SETDASH',82.)
CALL USET ('RADIANS')
DO I I = 1, 11
  X = (VX*I) + X8
  Y = T + (VX - (18.*I)) * Y8
  DYDX = 1 - (32 * (X-X8) / VX**2)
  CALL UCOSYS (X,Y,1.,1.,ATAN(DYDX))
  CALL UPEN1 (8.,8.,'DASH')
  CALL UNINDO (8.,1.,-1.,8.)
  CALL USET ('RELATIVE')
  CALL UPEN1 (.85,8.,'LARROW')
  CALL UMOVE (-.85,8.)
  CALL UPEN1 (8.,-.85,'LARROW')
  CALL UMOVE (8.,.85)
  CALL USET ('ABSOLUTE')
  CALL UNINDO (-225.,225.,-225.,225.)
  T = T + .4758
I CONTINUE
CALL UEND
STOP
END

```

## CHAPTER IX

### THREE DIMENSIONAL GRAPHICS

The three dimensional version of GCS gives to the user the ability to describing his environment in three dimensions. Each of the primary two dimensional GCS routines have a three dimensional counterpart: U3PEN, U3MOVE, U3DRAW, U3PRNT, U3WRIT, U3CSYS, and U3WHER. The coordinates used by each of these routines may be specified as (X,Y,Z) rectangular coordinates, (R,O,Z) cylindrical coordinates, or (R,O, $\phi$ ) spherical coordinates. Use of the 3-D routines does not preclude use of the 2-D routines; the 2-D routines are defined so as to draw parallel to the current X-Y plane on a plane specified by a default Z coordinate, contained in the Graphics Status Area (GSA). This Z value may be set by the user with an UPSET ('ZVALUE', value) call. The default ZVALUE is zero. By using both 2-D and 3-D routines an image may be described in 3-space.

In order to specify the appearance of the object being drawn, the location of the viewer and the direction in which he is looking must be specified. By calling UVIEW (XVIEW,YVIEW,ZVIEW,XSITE,YSITE,ZSITE), the user can specify where the view point is located and at what point in the environment he is looking (view site). The environment will then be displayed from this viewpoint with the view site located on a line which passes through the center of the viewport. The viewport indicates the boundaries of the user's field of view and also how far the viewport is from the view point or the viewsite. The X-coordinates of the viewport will range in the interval  $(-A/2, A/2)$  and the Y-coordinates in the interval  $(-B/2, B/2)$ . Default values for UVIEW are (0,0,150,0,0,0) and for UVWPRT are (200,200,0). GCS will also automatically clip points outside of the viewing pyramid (behind the view point or outside the viewport). The user can select whether the resulting image will be mapped to the screen with a PERSPECTIVE (default) or ORTHOGONAL projection.

The user may specify which portion of this viewport will be displayed on the screen by setting his UWINDO. The UWINDO boundaries may or may not overlap the viewport. If no portion of the UWINDO corresponds to the viewport, the viewport will be expanded to encompass the UWINDO if the UWINDO changes. If the viewport is specified smaller than the UWINDO, the UWINDO will contract to encompass the viewport.

As in 2-D GCS, the user may construct arbitrary user coordinate systems using the calls U3CSYS, U3SCAL, and U3ROTA. These work similar to the 2-D case with one exception. Since 3-D rotations are not commutative, the order of rotation must be specified with a suitable USET call: 'XYZ', 'XZY', 'YXZ', 'ZXY', or 'ZYX'. The default sequence is 'ZYX'.

A call to the 2-D routine UCOSYS will cause a rotation about the Z axis. Any rotation around other than the system Z axis will cause the XY plane to cease being parallel to the screen and will effect the output of symbols and curves (see below).

Curve generation using the UARC, UCRCL, UCONIC, UPLYGN, and URECT routines function as they did before. However, since they are 2-D routines, their output will be created in the XY plane specified by the current ZVALUE parameter as indicated above. In addition, software characters generated by U3PRNT, U3WRIT, and UAOUT will also appear in the XY plane. Hardware characters still appear on the plane of the display surface. Line terminators (arrowheads and software characters) and tic marks appear on a plane specified by the two end points of the line and a third point which the user can specify via the UPOINT(X,Y,Z) subroutine. The default UPOINT is a point on the default XY-plane outside the default UWINDO boundaries. This point is (-.6931471806, -1.096122887, 0).

Textual output for three dimensions has been expanded slightly to allow another text output mode. This mode is 'XYZCOORDINATES' which will print the three components



of a set of 3-D coordinates. 'XYCOORDINATES' are still available for 2-D coordinates. All other text functions of 2-D GCS are processed as before. Since the text created by U3PRNT and U3WRIT is displayed in the current user X-Y plane, specified by the Z-coordinate of the specified location, suitable choices of UVIEW will result in text which when displayed will be backwards. This is a result of viewing the text from behind it.

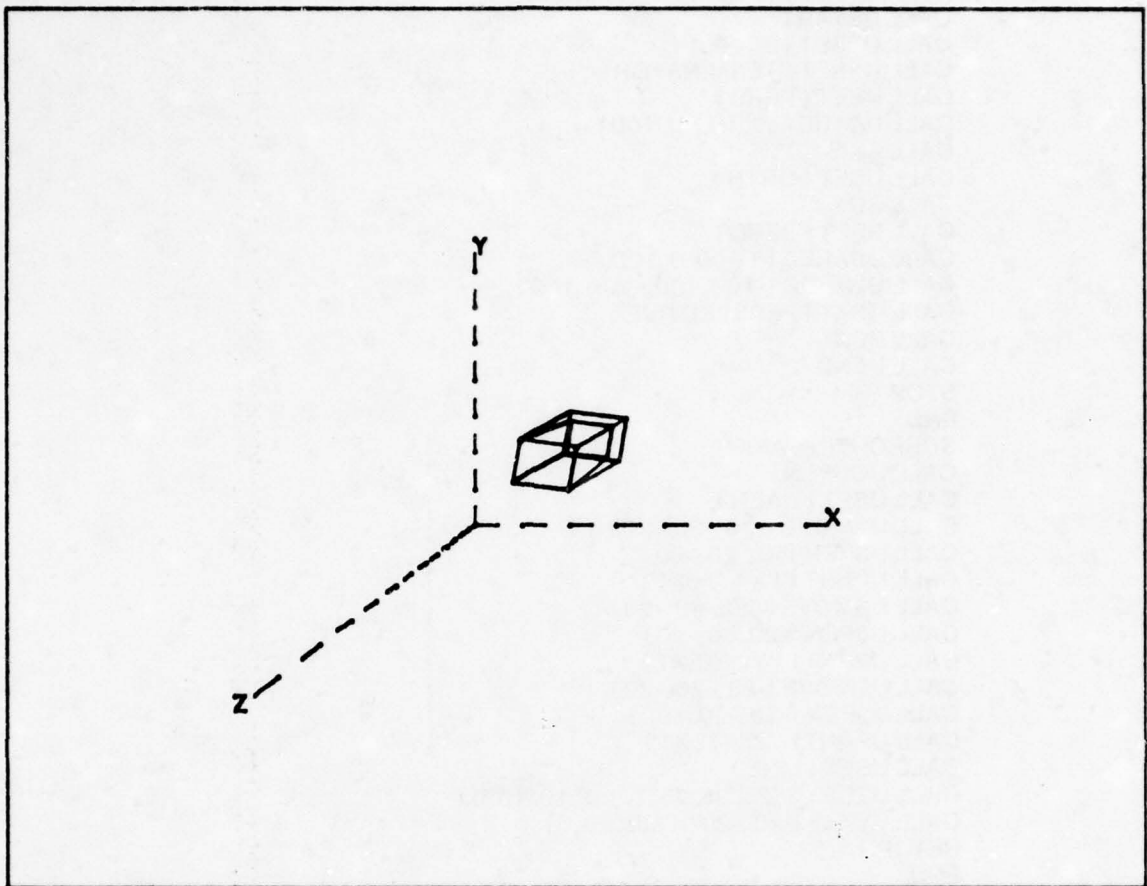
Drawing 3-D images is primarily intended for use in VIRTUAL space. However, a user who wishes to do so can directly address the face of the display surface in DEVICE space. If this is done, no clipping against viewport boundaries takes place. The requested graphical data is displayed projected orthogonally onto the device screen. For most devices, the Z-value will be ignored and the X,Y components will be used as screen addresses after conversion to device units.

To facilitate displaying different sections of the viewport, an additional window and display area routine has been implemented. ULOOK lets the user specify a portion of the display surface (UDAREA) in device units. The window (UWINDO) dimensions are then adjusted to move the UWINDO to display the equivalent portion of virtual space (viewport). The effect is that of moving a template around on the viewport.

C THIS PROGRAM DEMONSTRATES THE USE OF SEVERAL 3-D SUBROUTINES  
C AND THE DIFFERENCE BETWEEN ORTHOGONAL AND PERSEPCTIVE  
C PLOTTING.

```
CALL USTART
CALL UPSET ('SETD',1)
CALL UPSET ('TERMINATOR',',')
CALL USET ('PERC')
CALL UDAREA (0.,100.,0.,100.)
CALL AXIS
CALL USET ('ORTH')
CALL BOX
CALL USET ('PERC')
CALL UDAREA (0.,100.,0.,100.)
CALL UWINDO (-100.,100.,-100.,100.)
CALL USET ('PERSPECTIVE')
CALL BOX
CALL UEND
STOP
END
SUBROUTINE AXIS
CALL UOUTLN
CALL USET ('DASH')
CALL U3MOVE (-20.,-20.,-20.)
CALL U3PEN (50.,-20.,-20.)
CALL UPRNT1 ('X;', 'TEXT')
CALL U3MOVE (-20.,-20.,-20.)
CALL U3PEN (-20.,50.,-20.)
CALL UPRNT1 ('Y;', 'TEXT')
CALL U3MOVE (-20.,-20.,-20.)
CALL U3PEN (-20.,-20.,100.)
CALL UPRNT1 ('Z;', 'TEXT')
CALL USET ('LINE')
CALL U3CSYS (25.,25.,25.,1.,1.,1.,10.,10.,0.)
CALL UVIEW (-20.,-20.,-150.,0.,0.,0.)
RETURN
END
SUBROUTINE BOX
CALL U3MOVE (0.,0.,0.)
CALL U3PEN (10.,0.,0.)
CALL U3PEN (10.,10.,0.)
CALL U3PEN (0.,10.,0.)
CALL U3PEN 0.,0.,0.)
CALL U3MOVE (10.,0.,0.)
CALL U3PEN (10.,0.,60.)
CALL U3PEN (10.,10.,60.)
CALL U3PEN (10.,10.,0.)
CALL U3MOVE (0.,10.,0.)
CALL U3PEN (0.,10.,60.)
CALL U3PEN (10.,10.,60.)
CALL U3MOVE (0.,0.,0.)
CALL U3PEN (0.,0.,60.)
CALL U3PEN (0.,10.,60.)
CALL U3PEN (0.,10.,0.)
CALL U3MOVE (0.,0.,60.)
CALL U3PEN (10.,0.,60.)
RETURN
END
```

EXAMPLE IX-1



```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UPSET ('SPEED',128.)
CALL USET ('PERCENTUNITS')
CALL UDAREA (8.,188.,8.,188.)
CALL UWINDO (-188.,188.,-188.,188.)
CALL UERASE
CALL AXIS
CALL USET ('ORTHOGONAL')
CALL BOX
CALL UDAREA (8.,188.,8.,188.)
CALL UWINDO (-188.,188.,-188.,188.)
CALL USET ('PERSPECTIVE')
CALL BOX
CALL UEND
STOP
END

```



```

SUBROUTINE AXIS
CALL UOUTLN
CALL USET ('DASH')
CALL USMOVE (-20.,-20.,-20.)
CALL USPEN (50.,-20.,-20.)
CALL UPRNT1 ('X','TEXT')
CALL USMOVE (-20.,-20.,-20.)
CALL USPEN (-20.,50.,-20.)
CALL UPRNT1 ('Y','TEXT')
CALL USMOVE (-20.,-20.,-20.)
CALL USPEN (-20.,-20.,100.)
CALL UPRNT1 ('Z','TEXT')
CALL USET ('LINE')
CALL USCSYS (25.,25.,25.,1.,1.,1.,10.,10.,0.)
CALL UVIEW (-20.,-20.,-150.,0.,0.,0.)
RETURN
END

```

```

SUBROUTINE BOX
CALL USMOVE (0.,0.,0.)
CALL USPEN (10.,0.,0.)
CALL USPEN (10.,10.,0.)
CALL USPEN (0.,10.,0.)
CALL USPEN (0.,0.,0.)
CALL USMOVE (10.,0.,0.)
CALL USPEN (10.,0.,00.)
CALL USPEN (10.,10.,00.)
CALL USPEN (10.,10.,0.)
CALL USMOVE (0.,10.,0.)
CALL USPEN (0.,10.,00.)
CALL USPEN (10.,10.,00.)
CALL USMOVE (0.,0.,0.)
CALL USPEN (0.,0.,00.)
CALL USPEN (0.,10.,00.)
CALL USPEN (0.,10.,0.)
CALL USMOVE (0.,0.,00.)
CALL USPEN (10.,0.,00.)
RETURN
END

```

C THIS PROGRAM DEMONSTRATES THE USE OF SEVERAL 3-D SUBROUTINES  
C TO PRODUCE A VILLAGE ON A TEKTRONIX 4014/4015 TERMINAL.  
C

```
CALL USTART
CALL USET ('VIEW')
CALL UVWPRT (150.)
CALL UPSET ('TERM', '<')
CALL UWINDO (-100.,100.,-100.,100.)
CALL UDAREA (0.,7.,0.,7.)
CALL UVIEW (-40.,200.,700.,-20.,20.,0.)
CALL VILLAG
CALL UDAREA (7.1,14.1,0.,7.)
CALL UVIEW (-70.,-150.,50.,0.,0.,10.)
CALL VILLAG
CALL UEND
STOP
END
SUBROUTINE VILLAG
CALL USET ('XYZ')
CALL USET ('SYST')
CALL USET ('REFE')
CALL U3CSYS (-50.,20.,0.,1.,1.,1.,90.,0.,0.)
CALL USET ('BLACK')
CALL CHURCH
CALL USET ('SYST')
CALL U3CSYS (24.,-19.,0.,1.,1.,1.,80.,-90.,0.)
CALL USET ('RED')
CALL SCHOOL
CALL USET ('SYST')
CALL U3CSYS (70.,70.,0.,0.7,0.7,0.7,0.,0.,0.)
CALL USET ('BLUE')
CALL PIZZA
CALL USET ('SYST')
CALL U3CSYS (0.,0.,0.,1.,1.,1.,0.,0.,0.)
CALL USET ('BLACK')
CALL ROAD
CALL USET ('SYST')
RETURN
END
SUBROUTINE SCHOOL
CALL USET ('SOFT')
CALL USET ('REFE')
CALL U3MOVE (0.,0.,0.)
CALL URECT (50.,25.)
CALL U3MOVE 3.,4.,0.)
CALL URECT (7.,8.)
CALL U3MOVE (13.,4.,0.)
CALL URECT (17.,8.)
CALL U3MOVE (47.,4.,0.)
CALL URECT (43.,8.)
CALL U3MOVE (37.,4.,0.)
CALL URECT (33.,8.)
CALL U3MOVE (25.,0.,0.)
CALL URECT (22.,8.)
CALL URECT (28.,8.)
DO 10 I=3,43,10
X=I
CALL U3MOVE (X,15.,0.)
```

```

10 CALL URECT (X+4.,20.)
   CALL USET ('WORK')
   CALL U3CSYS (50.,0.,0.,1.,1.,1.,0.,90.,0.)
   ISW=1
15 CALL U3MOVE (0.,0.,0.)
   CALL URECT (90.,25.)
   DO 20 I=3,83,10
   X=1
   CALL U3MOVE (X,16.,0.)
   CALL URECT (X+4.,20.)
   IF (I.EQ.43) GO TO 20
   CALL U3MOVE (X,4.,0.)
   CALL URECT (X+4.,8.)
20 CONTINUE
   CALL U3MOVE (45.,0.,0.)
   CALL URECT (42.,8.)
   CALL URECT (48.,8.)
   IF ISW.EQ.2) GO TO 30
   ISW=2
   CALL U3CSYS (0.,0.,-90./1.,1.,1.,0.,-90.,0.)
   GO TO 15
30 CALL UPSET ('VERT',3.)
   CALL UPSET ('HORI',3.)
   CALL UPRINT (0.,21.,5.'ROOSEVELT ELEMENTARY SCHOOL <')
   CALL USET ('REFE')
   CALL UPSET ('ZVAL',-90.)
   CALL U3MOVE (0.,0.,-90.)
   CALL URECT (50.,25.)
   DO 40 I=3,43,10
   X=1
   CALL U3MOVE (X-4.,-80.)
   CALL URECT (X+4.,8.)
   CALL U3MOVE (X,16.,-90.)
40 CALL URECT (X+4.,20.)
   CALL UPSET ('ZVAL',0.)
   RETURN
   END
   SUBROUTINE CHURCH
   CALL USET ('REFE')
   CALL U3MOVE (0.,0.,0.)
   CALL URECT (30.,40.)
   CALL U3MOVE (15.,0.,0.)
   CALL URECT (10.,10.)
   CALL URECT (20.,10.)
   CALL U3MOVE (14.,3.,0.)
   CALL URECT (14.25,6.)
   CALL U3MOVE (15.,75,3.,0.)
   CALL URECT (16.,6)
   CALL U3MOVE (0.,40.,0.)
   CALL U3PNE (30.,40.,0.)
   CALL USET ('WORK')
   CALL U3CSYS (30.,0.,0.,1.,1.,1.,0.,90.,0.)
   ISW=1
10 CALL U3MOVE (0.,0.,0.)
   CALL URECT (80.,40.)
   DO 20 I=20,40,10
   X=1
   CALL U3MOVE (X,10.,0.)

```



```

CALL U3PEN (X,25.,0.)
CALL U3PEN (X+4.,30.,0.)
CALL U3PEN (X+8.,25.,0.)
CALL U3PEN (X+8.,10.,0.)
20 CALL U3PEN (X,10.,0.)
IF (ISW,EQ.2) GO TO 30
ISW = 2
CALL U3CSYS (0.,0.,0.,1.,1.,1.,0.,90.,0.)
GO TO 10
30 CALL USET ('REFE')
CALL U3MOVE (0.,40.,-80.)
CALL U3PEN (15.,47.5,-80.)
CALL U3PEN (30.,40.,-80.)
CALL U3MOVE (0.,0.,-80.)
CALL E3PEN (30.,0.,-80.)
CALL U3MOVE (15.,47.5,-80.)
CALL U3PEN (15.,47.5,0.)
CALL U3MOVE (10.,45.,0.)
CALL U3PEN (10.,45.,-10.)
CALL U3PEN (15.,47.5,-10.)
CALL U3PEN (20.,45.,-10.)
CALL U3PEN (20.,45.,0.)
CALL U3PEN (15.,100.,-5.)
CALL U3PEN (10.,45.,0.)
CALL U3MOVE (10.,45.,-10.)
CALL U3PEN (15.,100.,-5.)
CALL U3PEN (20.,45.,-10.)
CALL U3MOVE (15.,100.,-5.)
CALL U3PEN (15.,110.,-5.)
CALL U3MOVE (13.,107.,-5.)
CALL U3PEN (17.,107.,-5.)
RETURN
END
SUBROUTINE ROAD
CALL U3MOVE (-200.,9.,0.)
CALL U3PEN (-9.,9.,0.)
CALL U3PEN (-9.,200.,0.)
CALL U3MOVE (200.,9.,0.)
CALL U3PEN (9.,9.,0.)
CALL U3PEN (9.,200.,0.)
CALL U3MOVE (-200.,-9.,0.)
CALL U3PEN (-9.,-9.,0.)
CALL U3PEN (-9.,-200.,0.)
CALL UMOVE (9.,-200.,0.)
CALL U3PEN (9.,-9.,0.)
CALL U3PEN (200.,-9.,0.)
RETURN
END
SUBROUTINE PIZZA
CALL USET ('SOFT')
CALL USET ('REFE')
CALL U3MOVE (0.,0.,0.)
CALL U3CSYS (0.,0.,0.,1.,1.,1.,90.,0.,0.)
CALL U3PEN (40.,0.,0.)
CALL U3PEN (20.,50.,0.)
CALL U3PEN (0.,0.,0.)
CALL U3MOVE (20.,0.,0.)
CALL URECT (17.5,8)

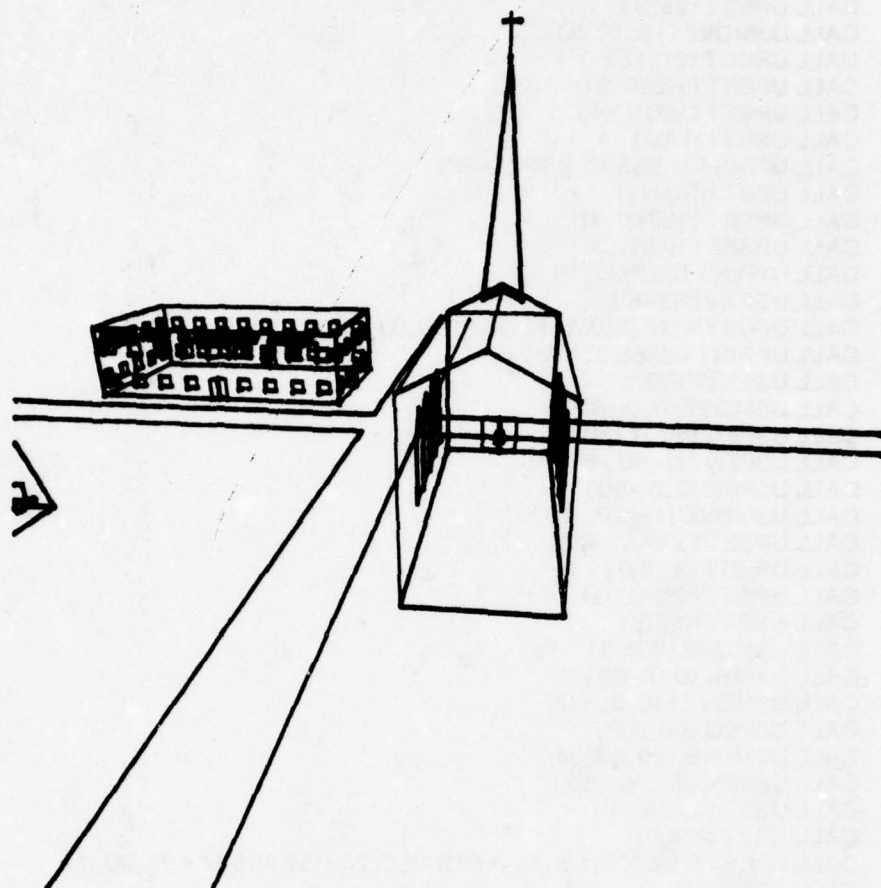
```

```

CALL URECT (22.5,8.)
CALL U3MOVE (10.,2.,0.)
CALL URECT (15.,8.)
CALL U3MOVE (30.,2.,0.)
CALL URECT (25.,8.)
CALL U3MOVE (13.,10.,0.)
CALL URECT (27.,15.)
CALL UPSET ('HORI',2.)
CALL UPSET ('VERT', 4.)
CALL USET ('ITAL')
CALL UPRINT (15.65,11.2,'PIZZA<')
CALL USET ('GOTH')
CALL UPSET ('VERT',.4)
CALL UPSET ('HORI',.3)
CALL UPRINT (20.95,5.,'IN<')
CALL USET ('WORK')
CALL U3CSYS (40.,0.,0.,1.,1.,1.,0.,-180.,0.)
CALL UPRINT (20.65,5.,'OUT<')
CALL USET ('REFE')
CALL U3MOVE (0.,0.,-60.)
CALL U3PEN (40.,0.,-60.)
CALL U3PEN (20.,50.,-60.)
CALL U3PEN (0.,0.,-60.)
CALL U3MOVE (18.5,0.,-60.)
CALL UPSET ('ZVAL', -60.)
CALL URECT (21.5,8.)
CALL UPSET ('ZVAL',0.)
CALL USET ('REFE')
CALL U3MOVE (0.,0.,0.)
CALL U3PEN (0.,0.,-60.)
CALL U3MOVE (40.,0.,-60.)
CALL U3PEN (40.,0.,0.)
CALL U3MOVE (20.,50.,0.)
CALL U3PEN (20.,50.,-60.)
CALL USET ('WORK')
CALL USET ('XYZ')
CALL U3CSYS (0.,0.,0.,1.,1.,1.,-ATAN2(50.,20.)*57.29577+90.,90.,0.)
CALL USET ('XYZ')
CALL UPSET ('HORI',2.)
CALL UPSET ('VERT',2.5)
CALL UPRINT (5.,5.,'GIANT PIZZA $8<')
CALL USET ('REFE')
RETURN
END

```

#### EXAMPLE IX-2



```

CALL USTART
CALL USET ('VIEWDISTANCE')
CALL UVMPT (100.)
CALL UPSET ('TERMINATOR',',',',')
CALL UWINDO (-100.,100.,-100.,100.)
CALL UDAREA (0.,10.0,0.,10.0)
CALL UVIEW (-40.,200.,70.,-20.,20.,0.)
CALL VILLAS
CALL UEND
STOP
END

```



```

SUBROUTINE VILLAS
CALL USET ('XYZ ')
CALL USET ('SYSTEMAXIS')
CALL USET ('REFERENCEAXIS')
CALL USCSYS (-58.,28.,8.,1.,1.,1.,98.,8.,8.)
CALL USET ('BLACK')
CALL CHURCH
CALL USET ('SYSTEMAXIS')
CALL USCSYS (24.,-18.,8.,1.,1.,1.,98.,-98.,8.)
CALL USET ('RED ')
CALL SCHOOL
CALL USET ('SYSTEMAXIS')
CALL USCSYS (78.,78.,8.,8.7,8.7,8.7,8.,8.,8.)
CALL USET ('BLUE')
CALL PIZZA
CALL USET ('SYSTEMAXIS')
CALL USCSYS (8.,8.,8.,1.,1.,1.,9.,8.,8.)
CALL USET ('BLACK')
CALL ROAD
CALL USET ('SYSTEMAXIS')
RETURN
END

```

```

SUBROUTINE SCHOOL
CALL USET ('SOFT')
CALL USET ('REFERENCEAXIS')
CALL USMOVE (8.,8.,8.)
CALL URECT (58.,25.)
CALL USMOVE (3.,4.,8.)
CALL URECT (7.,8.)
CALL USMOVE (19.,4.,8.)
CALL URECT (17.,8.)
CALL USMOVE (47.,4.,8.)
CALL URECT (48.,8.)
CALL USMOVE (37.,4.,8.)
CALL URECT (38.,8.)
CALL USMOVE (25.,8.,8.)
CALL URECT (22.,8.)
CALL URECT (28.,8.)
DO 18 I = 3, 48, 18
X = I
CALL USMOVE (X,18.,8.)
18 CALL URECT (X+4.,28.)
CALL USET ('WORKINGAXIS')
CALL USCSYS (58.,8.,8.,1.,1.,1.,8.,98.,8.)
ISN = 1
15 CALL USMOVE (8.,8.,8.)
CALL URECT (98.,25.)
DO 28 I = 3, 63, 18
X = I
CALL USMOVE (X,18.,8.)
CALL URECT (X+4.,28.)

```

```

IF (I.EQ. 49) GO TO 28
CALL USMOVE (X,4.,8.)
CALL URECT (X+4.,8.)
28 CONTINUE
CALL USMOVE (45.,8.,8.)
CALL URECT (42.,8.)
CALL URECT (48.,8.)
IF (ISN.EQ. 2) GO TO 38
ISN = 2
CALL USCSYS (8.,8.,-88.,1.,1.,1.,8.,-88.,8.)
GO TO 15
38 CALL UPSET ('VERTICAL',9.)
CALL UPSET ('HORIZONTAL',9.)
CALL UPRINT (8.,21.5,'ROOSEVELT ELEMENTARY SCHOOL,')
CALL USET ('REFERENCEAXIS')
CALL UPSET ('ZVALUE',-88.)
CALL USMOVE (8.,8.,-88.)
CALL URECT (58.,25.)
DO 48 I = 3, 49, 18
X = I
CALL USMOVE (X,4.,-88.)
CALL URECT (X+4.,8.)
CALL USMOVE (X,18.,-88.)
48 CALL URECT (X+4.,28.)
CALL UPSET ('ZVALUE',8.)
RETURN
END

```

```

SUBROUTINE CHURCH
CALL USET ('REFERENCEAXIS')
CALL USMOVE (8.,8.,8.)
CALL URECT (38.,48.)
CALL USMOVE (15.,8.,8.)
CALL URECT (18.,18.)
CALL URECT (28.,18.)
CALL USMOVE (14.,9.,8.)
CALL URECT (14.25,8.)
CALL USMOVE (15.75,9.,8.)
CALL URECT (18.,8.)
CALL USMOVE (8.,48.,8.)
CALL USPEN (15.,47.5,8.)
CALL USPEN (38.,48.,8.)
CALL USET ('WORKINGAXIS')
CALL USCSYS (38.,8.,8.,1.,1.,1.,8.,88.,8.)
ISN = 1
18 CALL USMOVE (8.,8.,8.)
CALL URECT (38.,48.)
DO 28 I = 28, 58, 18
X = I
CALL USMOVE (X,18.,8.)
CALL USPEN (X,25.,8.)
CALL USPEN (X+4.,38.,8.)
CALL USPEN (X+8.,25.,8.)
CALL USPEN (X+8.,18.,8.)
28 CALL USPEN (X,18.,8.)
IF (ISN.EQ. 2) GO TO 38

```

```

ISN = 2
CALL USCSYS (8.,8.,8.,1.,1.,1.,8.,98.,8.)
GO TO 18
38 CALL USET ('REFERENCEAXIS')
CALL USMOVE (8.,48.,-88.)
CALL USPEN (15.,47.5,-88.)
CALL USPEN (38.,48.,-88.)
CALL USMOVE (8.,8.,-88.)
CALL USPEN (38.,8.,-88.)
CALL USMOVE (15.,47.5,-88.)
CALL USPEN (15.,47.5,8.)
CALL USMOVE (18.,45.,8.)
CALL USPEN (18.,45.,-18.)
CALL USPEN (15.,47.5,-18.)
CALL USPEN (28.,45.,-18.)
CALL USPEN (28.,45.,8.)
CALL USPEN (15.,188.,-5.)
CALL USPEN (18.,45.,8.)
CALL USMOVE (18.,45.,-18.)
CALL USPEN (15.,188.,-5.)
CALL USPEN (28.,45.,-18.)
CALL USMOVE (15.,188.,-5.)
CALL USPEN (15.,118.,-5.)
CALL USMOVE (19.,187.,-5.)
CALL USPEN (17.,187.,-5.)
RETURN
END

```

```

SUBROUTINE ROAD
CALL USMOVE (-228.,8.,8.)
CALL USPEN (-8.,8.,8.)
CALL USPEN (-8.,228.,8.)
CALL USMOVE (228.,8.,8.)
CALL USPEN (8.,8.,8.)
CALL USPEN (8.,228.,8.)
CALL USMOVE (-228.,-8.,8.)
CALL USPEN (-8.,-8.,8.)
CALL USPEN (-8.,-228.,8.)
CALL USMOVE (8.,-228.,8.)
CALL USPEN (8.,-8.,8.)
CALL USPEN (228.,-8.,8.)
RETURN
END

```



```

SUBROUTINE PIZZA
CALL USET ('SOFTWARE CHARACTERS')
CALL USET ('REFERENCEAXIS')
CALL USMOVE (8.,8.,8.)
CALL USCYS (8.,8.,8.,1.,1.,1.,88.,8.,8.)
CALL USPEN (48.,8.,8.)
CALL USPEN (28.,58.,8.)
CALL USPEN (8.,8.,8.)
CALL USMOVE (28.,8.,8.)
CALL URECT (17.5,8.)
CALL URECT (22.5,8.)
CALL USMOVE (18.,2.,8.)
CALL URECT (15.,8.)
CALL USMOVE (38.,2.,8.)
CALL URECT (25.,8.)
CALL USMOVE (19.,18.,8.)
CALL URECT (27.,15.)
CALL UPSET ('HORIZONTAL',2.)
CALL UPSET ('VERTICAL',4.)
CALL USET ('ITALICS')
CALL UPRINT (15.85,11.2,'PIZZA,')
CALL USET ('GOTHIC')
CALL UPSET ('VERTICAL',4)
CALL UPSET ('HORIZONTAL',9)
CALL UPRINT (28.85,5.,'IN,')
CALL USET ('WORKINGAXIS')
CALL USCYS (48.,8.,8.,1.,1.,1.,8.,-188.,8.)

```

```

CALL UPRINT (28.85,5.,'OUT,')
CALL USET ('REFERENCEAXIS')
CALL USMOVE (8.,8.,-88.)
CALL USPEN (48.,8.,-88.)
CALL USPEN (28.,58.,-88.)
CALL USPEN (8.,8.,-88.)
CALL USMOVE (18.5,8.,-88.)
CALL UPSET ('ZVALUE',-88.)
CALL URECT (21.5,8.)
CALL UPSET ('ZVALUE',8.)
CALL USET ('REFERENCEAXIS')
CALL USMOVE (8.,8.,8.)
CALL USPEN (8.,8.,-88.)
CALL USMOVE (48.,8.,-88.)
CALL USPEN (48.,8.,8.)
CALL USMOVE (28.,58.,8.)
CALL USPEN (28.,58.,-88.)
CALL USET ('WORKINGAXIS')
CALL USET ('XYZ')
CALL USCYS (8.,8.,8.,1.,1.,1.,-ATAN2 (58.,28.)+57.29577+88.,
88.,8.)
CALL USET ('XYZ')
CALL UPSET ('HORIZONTAL',2.)
CALL UPSET ('VERTICAL',2.5)
CALL UPRINT (5.,5.,'GIANT PIZZA 88,')
CALL USET ('REFERENCEAXIS')
RETURN
END

```

## CHAPTER X

### GRAPHICAL DATA STRUCTURE PROCESSING

A feature which is contained within 3D GCS is the ability to create, save, and later recreate pictures as they are drawn. It permits the user to specify the description of a commonly used graphic image once, store it, and then invoke this description whenever necessary. This facility is the implementation of an internal high-level GCS data structure.

Before the user can define or use a data structure, he must first provide a work file for GCS to use. This work file can be specified by an UPSET ('LIBRARYFILE', file number) call. The file will then be used to maintain the user's currently active library of GCS graphics data structures.

To build a data structure, the user must first call a routine USTRCT (NAME) which initiates construction of a data structure under the specified name. From then on all calls to the GCS routines listed in Table VI-1 will be saved as elements of the data structure. Whenever the specification of the object has been completed, a call is made to UTERM (NAME) to terminate the building process. The resulting data structure is then stored on the library file. During the building process, the build mode may be turned off by specifying 'NOBUILD'. Construction may be resumed by then specifying 'BUILD'. 'BUILD' is automatically specified when USTRCT is called and 'NOBUILD' is set when UTERM is called. Normally, the structure being built will be displayed during the construction process. If the user does not wish to view the structure he may specify 'NOEXECUTE'. If this is done, the structure element will be saved but no visible output will appear. The user may return to viewing his construction process by setting 'EXECUTE' which is the default mode.

To invoke an already created structure, the user uses the U3CALL or UCALL routines specifying the name, position, and orientation of the data structure. The data structure specified will be displayed as indicated. If another structure is currently being built when this call takes place, a U3CALL structure element for the invoked structure will be inserted in the active structure.

When the user terminates the GCS program, he may wish to save his library file so it can be restored at a later time. A routine called UTILTY is used to perform several utility functions concerned with maintaining the library file. To save the current library file, the user simply invokes UTILTY ('SAVE', file number). This file number must be different from the library file. All data structure contained in the library will be reformatted into a standard Hollerith card format and written to the save file. To restore a saved file to the library file, the UTILTY ('LOAD', file number) function would be used. Other UTILTY functions exist to MERGE a save file into the existing library file, PURGE the existing library file, and DELETE or RENAME structures in the library file.

**TABLE X-1**

**DATA STRUCTURE ELEMENTS**

The following routines will cause elements to be inserted  
in GCS data structures being built

UARC  
UCALL/U3CALL\*  
UCOSYS/U3CSYS\*  
UCRCLE  
UDRAW/U3DRAW\*  
UMOVE/U3MOVE\*  
UPEN/U3PEN\*  
UPRINT/U3PRNT\*  
UPRNT1  
UPSET  
URECT  
UROTAT/U3ROTA\*  
USCALE/U3SCAL\*  
USET  
UWRITE/U3WRIT\*  
UWRIT1

\*2-D calls stored in 3-D form



C THIS PROGRAM DEMONSTRATES THE SAVING OF DATA STRUCTURES AND  
C SHOWS THE DIFFERENCE BETWEEN ORTHOGONAL AND PERSPECTIVE  
C PLOTTING.

C ATTACH A PERMANENT FILE TO SAVE THE DATA STRUCTURES ON A  
C HONEYWELL COMPUTER.

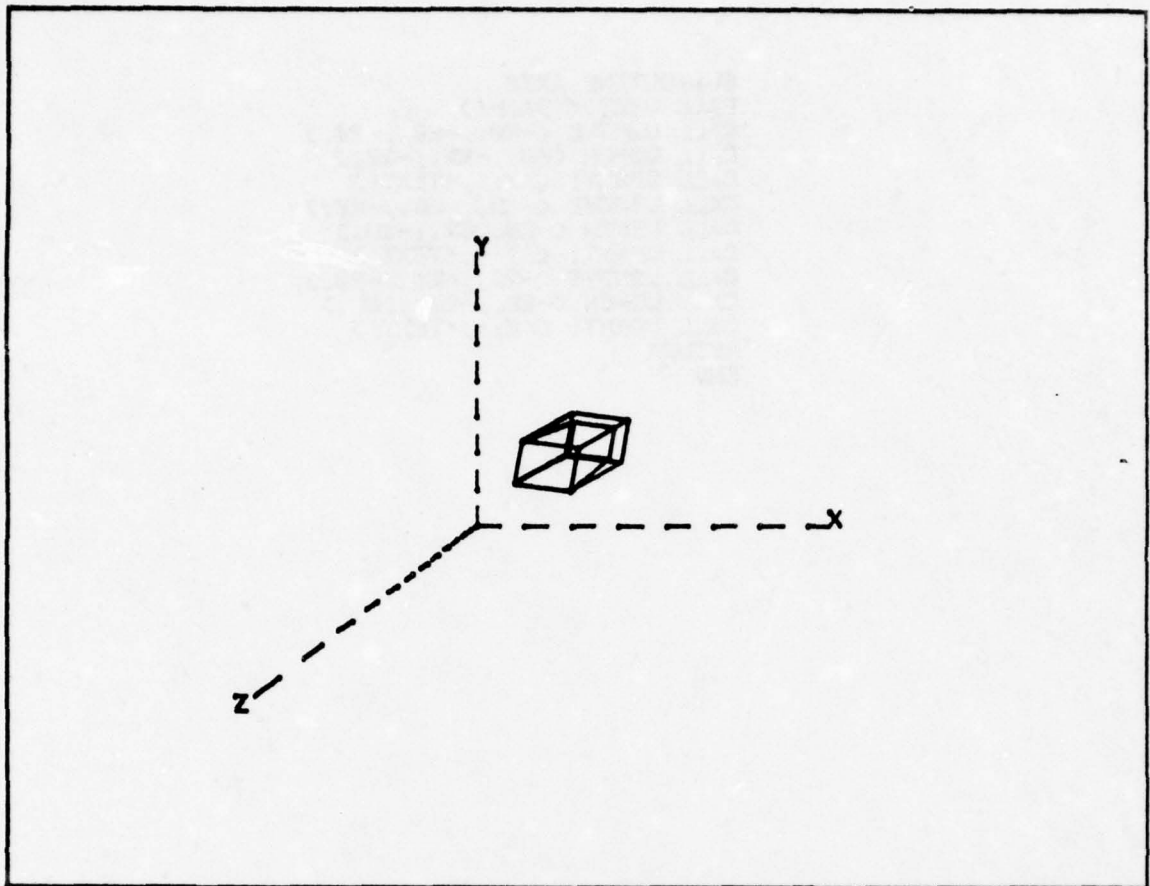
C  
CALL ATTACH (8,'/TEK3DEMO/SAVE',3,0,IST)  
CALL USTART  
CALL UPSET ('LIBR',1.)  
CALL UPSET ('SETD',1.)  
CALL UPSET ('TERMINATOR',';')  
CALL USET ('PERC')  
CALL UDAREA (0.,100.,0.,100.)  
CALL UOUTLN  
CALL UWINDO (-100.,100.,-100.,100.)  
CALL USTRCT ('AXIS')  
CALL AXIS  
CALL UTERM ('AXIS')  
CALL USET ('ORTH')  
CALL U3CSYS (25.,25.,25.,1.,1.,1.,10.,10.,0.)  
CALL UVIEW (-20.,-20.,-150.,0.,0.,0.)  
CALL USTRCT ('BOX')  
CALL BOX  
CALL UTERM ('BOX')  
CALL USET ('PERSPECTIVE')  
CALL BOX  
CALL UTILITY ('SAVE',8.)  
CALL UEND  
STOP  
END  
SUBROUTINE AXIS  
CALL USET ('DASH')

C  
C NEED TO DO A MOVE INSIDE STRUCTURE  
C

CALL U3MOVE (-20.,-20.,-20.)  
CALL U3PEN (50.,-20.,-20.)  
CALL UPRNT1 ('X;', 'TEXT')  
CALL U3MOVE (-20.,-20.,-20.)  
CALL U3PEN (-20.,50.,-20.)  
CALL UPRNT1 ('Y;', 'TEXT')  
CALL U3MOVE (-20.,-20.,-20.)  
CALL U3PEN (-20.,-20.,100.)  
CALL UPRNT1 ('Z;', 'TEXT')  
RETURN  
END  
SUBROUTINE BOX  
CALL USET ('LINE')  
CALL UMOVE (0.,0.,0.)  
CALL U3PEN (10.,0.,0.)  
CALL U3PEN (10.,10.,0.)  
CALL U3PEN (0.,10.,0.)  
CALL U3PEN (0.,0.,0.)  
CALL U3MOVE (10.,0.,0.)  
CALL U3PEN (10.,0.,60.)  
CALL U3PEN (10.,10.,60.)  
CALL U3PEN (10.,10.,0.)

```
CALL U3MOVE (0.,10.,0.)  
CALL U3PEN (0.,10.,60.)  
CALL U3PEN (10.,10.,60.)  
CALL U3MOVE (0.,0.,0.)  
CALL U3PEN (0.,0.,60.)  
CALL U3PEN (0.,10.,60.)  
CALL U3PEN (0.,10.,0.)  
CALL U3MOVE (0.,0.,60.)  
CALL U3PEN (10.,0.,60.)  
RETURN  
END
```

EXAMPLE X-1



```

CALL ATTACH (6, '/TEKSDemo/SAVE', '9,8,1ST,')
CALL USTART
CALL UPSET ('TERMINATOR', ',')
CALL UPSET ('SPEED', 128.)
CALL UPSET ('LIBRARY', 1.)
CALL UPSET ('SETDASH', 1.)
CALL USET ('PERCENTUNITS')
CALL UDAREA (8., 128., 8., 128.)
CALL UERASE
CALL UOUTLN
CALL UWINDO (-128., 128., -128., 128.)
CALL USTRUCT ('AXIS ')
CALL AXIS
CALL UTERM ('AXIS ')
CALL USET ('ORTHOGONAL')
CALL USCOSYS (25., 25., 25., 1., 1., 1., 18., 18., 8.)
CALL UVIEW (-25., -25., -158., 8., 8., 8.)
CALL USTRUCT ('BOX ')
CALL BOX
CALL UTERM ('BOX ')
CALL USET ('PERSPECTIVE')
CALL BOX
CALL UTILITY ('SAVE', 6.)
CALL UEND
STOP
END

```



```

SUBROUTINE AXIS
CALL USET ('DASH')
CALL USMOVE (-28.,-28.,-28.)
CALL USPEN (58.,-28.,-28.)
CALL UPNT1 ('X','TEXT')
CALL USMOVE (-28.,-28.,-28.)
CALL USPEN (-28.,58.,-28.)
CALL UPNT1 ('Y','TEXT')
CALL USMOVE (-28.,-28.,-28.)
CALL USPEN (-28.,-28.,188.)
CALL UPNT1 ('Z','TEXT')
RETURN
END

```

```

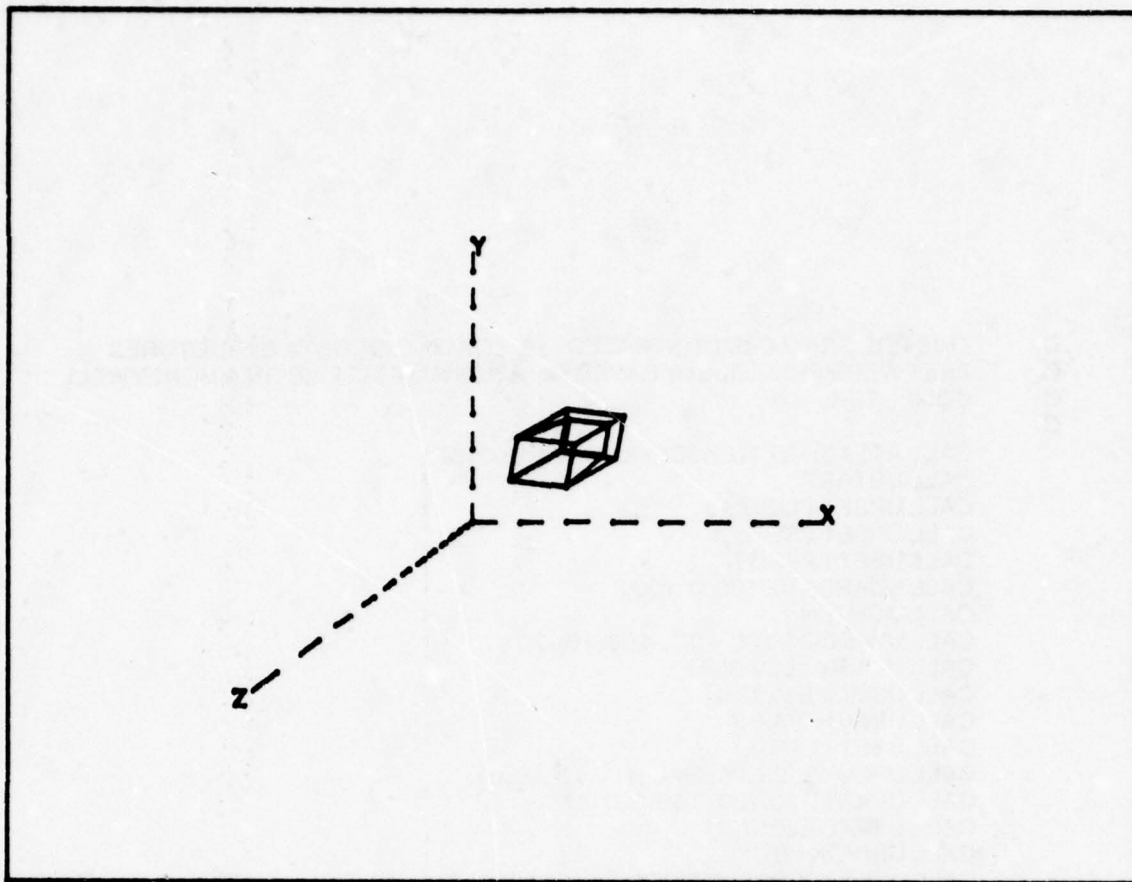
SUBROUTINE BOX
CALL USET ('LINE')
CALL USMOVE (8.,8.,8.)
CALL USPEN (18.,8.,8.)
CALL USPEN (18.,18.,8.)
CALL USPEN (8.,18.,8.)
CALL USPEN (8.,8.,8.)
CALL USMOVE (18.,8.,8.)
CALL USPEN (18.,8.,88.)
CALL USPEN (18.,18.,88.)
CALL USPEN (18.,18.,8.)
CALL USMOVE (8.,18.,8.)
CALL USPEN (8.,18.,88.)
CALL USPEN (18.,18.,88.)
CALL USMOVE (8.,8.,8.)
CALL USPEN (8.,8.,88.)
CALL USPEN (8.,18.,88.)
CALL USPEN (8.,18.,8.)
CALL USMOVE (8.,8.,88.)
CALL USPEN (18.,8.,88.)
RETURN
END

```

C THIS PROGRAM DEMONSTRATES THE LOADING OF DATA STRUCTURES  
C THAT WERE PREVIOUSLY SAVED ON A PERMANENT FILE ON A HONEYWELL  
C COMPUTER.

CALL ATTACH (8,'TEK3DEMO/SAVE;',3,0,IST.)  
CALL USTART  
CALL UPSET ('LIBR',1.)  
CALL UPSET ('SETD',1.)  
CALL USET ('PERC')  
CALL UDAREA (0.,100.,0.,100.)  
CALL UOUTLN  
CALL UWINDO (-100.,100.,-100.,100.)  
CALL UTILTY ('LOAD',8.)  
CALL U3MOVE (0.,0.,0.)  
CALL UINVOK ('AXIS')  
CALL USET ('ORTH')  
CALL U3CSYS (25.,25.,25.,1.,1.,1.,10.,10.,0.)  
CALL UVIEW (-20.,-20.,-150.,0.,0.,0.)  
CALL U3MOVE (0.,0.,0.)  
CALL UINVOK ('BOX')  
CALL USET ('PERSPECTIVE')  
CALL U3MOVE (0.,0.,0.)  
CALL UNIVOK ('BOX')  
CALL UEND  
STOP  
END

EXAMPLE X-2



```

CALL ATTACH (8, '/TENCIDENO/SAVE,', 8, 8, 127, 0)
CALL USTART
CALL UPSET ('LIBRARY', 1, 0)
CALL UPSET ('TERMINATOR', 1, 0)
CALL UPSET ('SPEED', 120, 0)
CALL USET ('PERCENTUNITS', 0)
CALL UDAREA (8, 100, 8, 100, 0)
CALL UERASE
CALL UOUTLN
CALL UNINDO (-100, 100, -100, 100, 0)
CALL UTILITY ('LOAD', 0, 0)
CALL USMOVE (8, 8, 8, 0)
CALL UINVOK ('AXIS', 0)
CALL USET ('ORTHOGONAL', 0)
CALL USCSYS (25, 25, 25, 1, 1, 1, 10, 10, 0, 0)
CALL UVIEW (-20, -20, -100, 8, 8, 8, 0)
CALL USMOVE (8, 8, 8, 0)
CALL UINVOK ('BOX', 0)
CALL USET ('PERSPECTIVE', 0)
CALL USMOVE (8, 8, 8, 0)
CALL UINVOK ('BOX', 0)
CALL UEND
STOP
END

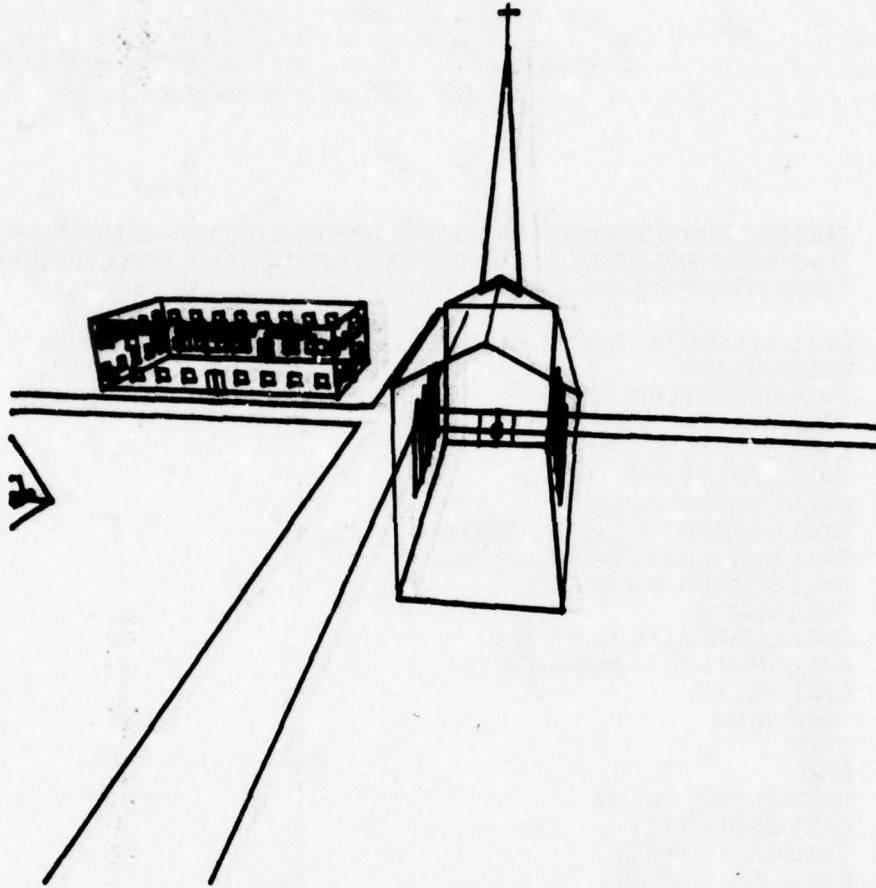
```



C THIS PROGRAM DEMONSTRATES THE LOADING OF DATA STRUCTURES  
C THAT WERE PREVIOUSLY SAVED ON A PERMANENT FILE ON A HONEYWELL  
C COMPUTER FOR DISPLAY ON TEKTRONIX 4014/4015 TERMINAL.

CALL ATTACH (8,'/TEK3DEMO/SAVE',3,0,IST)  
CALL USTART  
CALL UPSET ('LIBR',1)  
CALL UTILITY ('LOAD',8)  
CALL UPSET ('TERM','<')  
CALL USET ('VIEW')  
CALL UVWPRT (150)  
CALL UWINDO (-100,100,-100,100)  
CALL UVIEW (-40,200,70,-20,20,0)  
CALL UDAREA (0,7,0,7)  
CALL VILLAG  
CALL UDAREA (7.1,14,1,0,7)  
CALL VIEW (-70,-160,50,0,0,10)  
CALL VILLAG  
CALL UEND  
STOP  
END  
SUBROUTINE VILLAG  
CALL USET ('XYZ')  
CALL USET ('SYST')  
CALL USET ('REFE')  
CALL USET ('BLACK')  
CALL U3CALL (-50,20,0,1,1,1,90,0,0,'CHURCH')  
CALL USET ('RED')  
CALL U3CALL (24,-19,0,1,1,1,90,-90,0,'SCHOOL')  
CALL USET ('BLUE')  
CALL U3CALL (70,70,0,0,7,0,7,0,-7,0,0,0,'PIZZA')  
CALL USET ('BLACK')  
CALL U3CALL (0,0,0,1,1,1,0,0,0,'ROAD')  
RETURN  
END

EXAMPLE X-3



```

CALL ATTACH (8, '/TEXSDENO/SAVE', 'S,S,IST,.)
CALL USTART
CALL UPSET ('LIBRARY', 1.)
CALL UTILITY ('LOAD', 8.)
CALL UPSET ('TERMINATOR', ',,')
CALL USET ('VIEWDISTANCE')
CALL UVPRT (158.)
CALL UWINDO (-188., 188., -188., 188.)
CALL UVIEW (-48., 228., 78., -28., 28., 8.)
CALL VILLAS
CALL UEND
STOP
END
SUBROUTINE VILLAS
CALL USET ('XYZ')
CALL USET ('SYSTEMAXIS')
CALL USET ('REFERENCEAXIS')
CALL USET ('BLACK')
CALL USCALL (-58., 28., 8., 1., 1., 1., 88., 8., 8., 'CHURCH')
CALL USET ('RED')
CALL USCALL (24., -18., 8., 1., 1., 1., 88., -88., 8., 'SCHOOL')
CALL USET ('BLUE')
CALL USCALL (78., 78., 8., 8., 7., 8., 7., 8., 8., 8., 'PIZZA')
CALL USET ('BLACK')
CALL USCALL (8., 8., 8., 1., 1., 1., 8., 8., 8., 'ROAD')
RETURN
END

```

## CHAPTER XI

### PICTURE SEGMENTATION AND NAMING

#### Refresh Graphic Facilities

Many computer graphics display applications require only a static display in order to convey information. That is, the complete picture provides all of the relevant information that can be derived from the data used to produce the display. In some situations however, a static display provides incomplete information. For example, the plot of a missile trajectory does not reveal the relative speed of the projectile while in flight. In such situations, a dynamic display which portrays motion provides more information to the user than does a static display. GCS provides a facility for the dynamic display of information and for selective erase of information on refresh graphic display devices if available. The facility allows the user to identify and bracket a portion or portions of his display, assign a name to the section, turn the named section on or off, and replace a named section with a new definition of the section.

The basic unit of information of a dynamic display is the 'frame'. A frame represents a portion of a display having a known origin and a known termination. The GCS subroutines which delimit the start and the end of a frame are UFRAME and UFREND. They are invoked as follows:

CALL UFRAME (NAME)  
CALL UFREND (NAME)

where NAME is an eight character alphanumeric variable or constant. The subroutine UFRAME indicates the starting point of the framed information and subroutine UFREND denotes the ending point. Note that multiple frames may be defined, but nesting of frames is not permitted; that is, subroutine UFREND must be called to 'close' the current frame before the subroutine UFRAME is invoked to 'open' another frame.

Whenever a UFRAME/UFREND pair are invoked to replace the previous occurrence of the named frame, the previous occurrence is deleted upon completion of the new version of the frame, as signalled by the call to UFREND for that frame. Thus the complete frame will replace the previous complete frame. This is the default condition in GCS and is known as 'invisible' building of a frame. It can be specified by the following invocation to USET:

CALL USET ('INVISIBLE')

For some dynamic graphic applications, it is preferable to see the frame being built line-by-line. In this case, the previous frame must be deleted when the call to UFRAME is made for frame redefinition. The user can request the visible building of a frame by the following call:

CALL USET ('VISIBLE')

After a frame is completely built, and terminated by a call to UFREND, it may be selectively 'turned on' and 'turned off' by the use of the following GCS subroutines:

CALL USHOW (NAME)  
CALL UNSHOW (NAME)

where NAME is the eight character alphanumeric variable or constant which is the name assigned to the frame upon which the operation is to be performed. When USHOW is invoked for an invisible frame, the frame is made visible; if UNSHOW is invoked for a visible frame, the frame is made invisible.



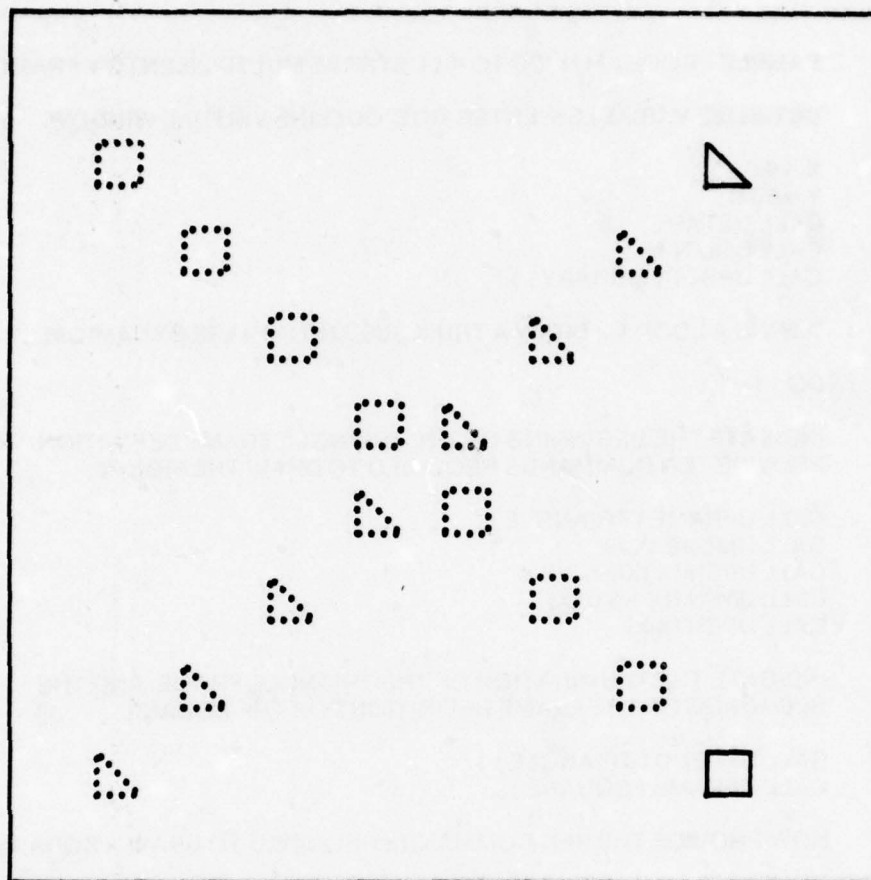
Note that the GCS subroutine UERASE affects the status of frames by completely removing all frames and frame names from GCS. Thus if USHOW or UNSHOW are invoked after a call to UERASE, an error will be indicated. In addition, it is improper to call UERASE while a frame is 'open', i.e., after a call to UFRAME and before a call to UFREND.

```

C   SAMPLE PROGRAM USED TO ILLUSTRATE MULTIPLE ENTITY FRAMING.
C
C   INITIALIZE VARIABLES, ENTER GCS, OUTLINE VIRTUAL WINDOW.
C
C   X=10.0
C   Y=80.0
C   CALL USTART
C   CALL UOUTLN
C   CALL UPSET ('LIBRARY', 1.)
C
C   DEFINE A LOOP TO DRAW A TRIANGLE AND SQUARE DYNAMICALLY.
C
C   DO 1 I=1,8
C
C   INDICATE THE BEGINNING OF THE TRIANGLE FRAME DEFINITION, THEN
C   PROVIDE TEN COMMANDS REQUIRED TO DRAW THE FIGURE.
C
C   CALL UFRAME ('TRIANGLE')
C   CALL UMOVE (X,X)
C   CALL UPEN (X,(X+5.0))
C   CALL UPEN ((X+5.0),X)
C   CALL UPEN (X,X)
C
C   INDICATE THE TERMINATION OF THE TRIANGLE FRAME, AND THE
C   BEGINNING OF THE FRAME DEFINITION FOR THE SQUARE.
C
C   CALL UFREND ('TRIANGLE')
C   CALL UFRAME ('SQUARE')
C
C   NOW PROVIDE THE PEN COMMANDS REQUIRED TO DRAW A SQUARE.
C
C   CALL UMOVE (X,Y)
C   X=X+10.0
C   Y=Y-10.0
C   CALL URECT ((X-5.0),Y+15.0))
C
C   INDICATE THE TERMINATION OF THE SQUARE FRAME, AND LOOP.
C
C   CALL UFREND ('SQUARE')
1  CONTINUE
C   CALL UEND
C   STOP
C   END

```

EXAMPLE XI-1



```

X = 18.8
Y = 88.8
CALL USTART
CALL UPSET ('LIBRARY',1.)
CALL UOUTLN
DO I = 1, 8
  CALL UFRAME ('TRIANGLE')
  CALL UMOVE (X,X)
  CALL UPEN (X,X+5.8))
  CALL UPEN (X+5.8),X)
  CALL UPEN (X,X)
  CALL UPEND ('TRIANGLE')
  CALL UFRAME ('SQUARE')
  CALL UMOVE (X,Y)
  X = X + 18.8
  Y = Y - 18.8
  CALL URECT (X-5.8),(Y+15.8))
  CALL UPEND ('SQUARE')
CONTINUE
CALL UEND
STOP
END

```

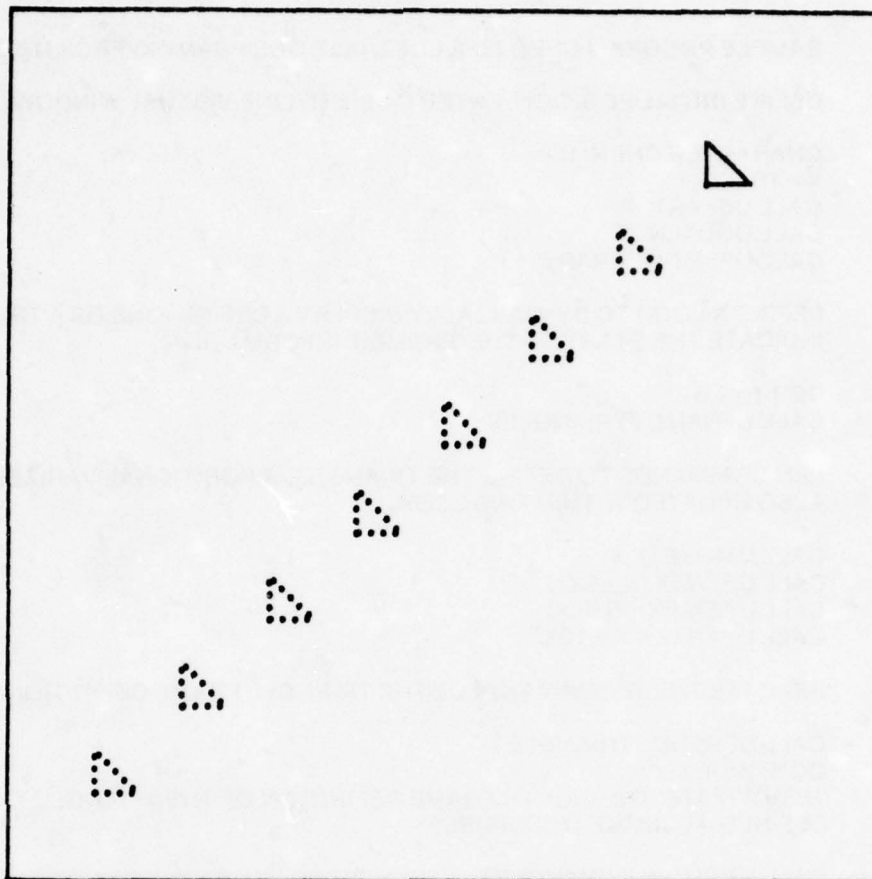


```

C   SAMPLE PROGRAM USED TO ILLUSTRATE GCS FRAMING FACILITIES.
C
C   DEFINE INITIAL POSITION, ENTER GCS, OUTLINE VIRTUAL WINDOW.
C
C   CHARACTER CHAR*1
C   X=10.0
C   CALL USTART
C   CALL UOUTLN
C   CALL UPSET ('LIBRARY',1.)
C
C   DEFINE A LOOP TO DYNAMICALLY DISPLAY 8 DEFINITIONS OF A TRIANGLE.
C   INDICATE THE START OF THE 'FRAMED' INFORMATION.
C
C   DO 1 I=1,8
C   CALL UFRAME ('TRIANGLE')
C
C   PEN COMMANDS TO DEFINE THE TRIANGLE. A POSITIONAL VARIABLE IS
C   ALSO UPDATED WITHIN THIS LOOP.
C
C   CALL UMOVE (X,X)
C   CALL UPEN (X,(X+5.0))
C   CALL UPEN ((X+5.0),X)
C   CALL UPEN (X=X+10.0)
C
C   INDICATE THE TERMINATION OF THE TRIANGLE FRAME DEFINITION.
C
C   CALL UFREND ('TRIANGLE')
C   1 CONTINUE
C   'DEACTIVATE' THE EIGHTH FRAME DEFINITION OF THE TRIANGLE THAT WAS
C   DEFINED; I.E., MAKE IT 'INVISIBLE'.
C
C   CALL UNSHOW ('TRIANGLE')
C
C   REQUEST A CHARACTER FROM THE KEYBOARD; ACTIVATE THE LAST
C   FRAME IF THE LETTER 'T' IS ENTERED; OTHERWISE, TERMINATE.
C
C   2 CALL UAIN (CHAR)
C   IF (CHAR.NE.'T') GO TO 3
C   CALL USHOW ('TRIANGLE')
C   GO TO 2
C   3 CALL UEND
C   STOP
C   END

```

#### EXAMPLE XI-2



```

CHARACTER CHAR=1
X=18.8
CALL USTART
CALL UPSET ('LIBRARY',1.)
CALL UOUTLN
DO 1 I = 1, 8
CALL UPRANE ('TRIANGLE')
CALL UMOVE (X,X)
CALL UPEN (X, X+5.8)
CALL UPEN (X+5.8, X)
CALL UPEN (X, X)
X = X + 18.8
CALL UPEND ('TRIANGLE')
1 CONTINUE
CALL USHOW ('TRIANGLE')
2 CALL UAIN (CHAR)
IF (CHAR.NE., 'T') GO TO 3
CALL USHOW ('TRIANGLE')
GO TO 2
3 CALL UEND
STOP
END

```

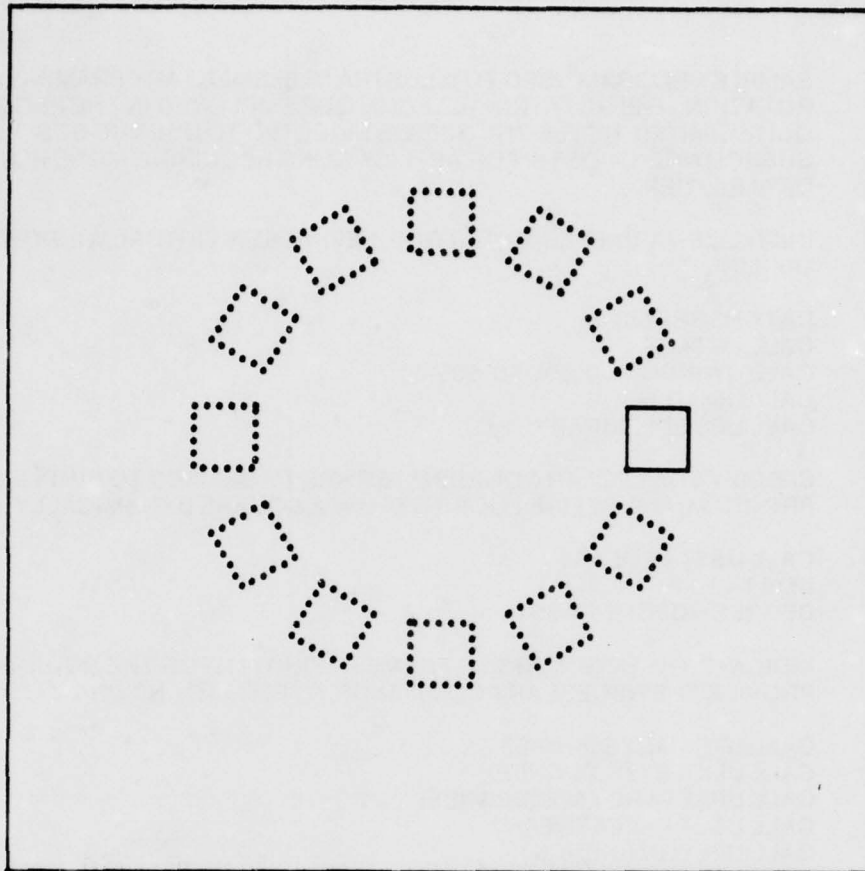
```

C      SAMPLE PROGRAM USED TO ILLUSTRATE ELEMENTARY FRAMING AND
C      ROTATION. THE ROTATIONAL TECHNIQUE EMPLOYED IN THE PROGRAM IS
C      QUITE LIMITED, HENCE THE USER IS DIRECTED TO USE THE GCS
C      SUBROUTINE 'UCOSYS' FOR APPLICATIONS REQUIRING MORE ROTATIONAL
C      CAPABILITIES.
C
C      INITIALIZE VARIABLES, ENTER GCS, DEFINE NEW VIRTUAL WINDOW, AND
C      OUTLINE IT.
C
C      DATA DEGREE/0.0/
C      CALL USTART
C      CALL UWINDO (-50.,50.,-50.,50.)
C      CALL UOUTLN
C      CALL UPSET ('LIBRARY', 1.)
C
C      SPECIFY THAT POLAR COORDINATES ARE TO BE USED TO SIMPLIFY THE
C      PROGRAM, AND DEFINE LOOP TO DRAW A SQUARE DYNAMICALLY.
C
C      CALL USET ('POLAR')
C      DO 1 I= 1,12
C      DEGREE = DEGREE + 30.0
C
C      INDICATE THE BEGINNING OF FRAME DEFINITION FOR THE SQUARE, AND
C      PROVIDE THE NECESSARY COMMANDS TO ROTATE AND DRAW IT.
C
C      CALL UFRAME ('SQUARE')
C      CALL UMOVE (25.,DEGREE)
C      CALL UPSET ('ROTATE',DEGREE)
C      CALL USET ('RELATIVE')
C      CALL UPLYGN (0.,0.,4.,5.)
C      CALL USET ('ABSOLUTE')
C
C      INDICATE THE TERMINATION OF THE SQUARE FRAME, AND THE LOOP.
C
C      CALL UFREND ('SQUARE')
1  CONTINUE
C      CALL UEND
C      STOP
C      END

```

#### EXAMPLE XI-3





```

DATA DEGREE/8.8/
CALL USTART
CALL UPSET ('LIBRARY',1.)
CALL UWINDO (-58.,58.,-58.,58.)
CALL UOUTLN
CALL USET ('POLAR')
DO I I = 1, 12
  DEGREE = DEGREE + 38.8
  CALL UFRAME ('SQUARE')
  CALL UMOVE (25.,DEGREE)
  CALL UPSET ('ROTATE',DEGREE)
  CALL USET ('RELATIVE')
  CALL UPLYON (8.,8.,4.,5.)
  CALL USET ('ABSOLUTE')
  CALL UPEND ('SQUARE')
CONTINUE
CALL UEND
STOP
END

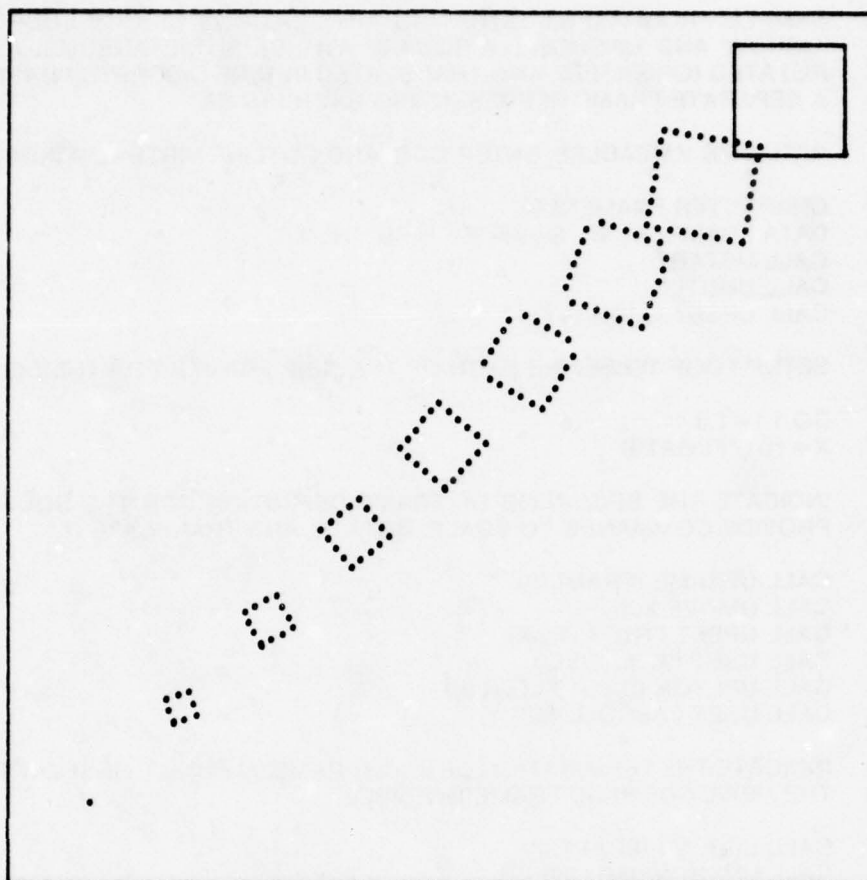
```

```

C   SAMPLE PROGRAM ILLUSTRATING APPLICATIONS OF GCS SUBROUTINES
C   'USHOW' AND 'UNSHOW'. A SQUARE WILL BE SIMULTANEOUSLY SCALED,
C   ROTATED (ORIENTED), AND TRANSLATED IN NINE DISCRETE PHASES, WITH
C   A SEPARATE FRAME REPRESENTING EACH PHASE.
C
C   INITIALIZE VARIABLES, ENTER GCS, AND OUTLINE VIRTUAL WINDOW.
C
C   CHARACTER FRAME*8(9)
C   DATA FRAME/'A','B','C','D','E','F','G','H','I'/
C   CALL USTART
C   CALL UOUTLN
C   CALL UPSET ('LIBRARY',1.)
C
C   SETUP LOOP TO DEFINE EACH OF THE NINE FRAMES FOR THE SQUARE.
C
C   DO 1 I=1,9
C   X=10.0*FLOAT(I)
C
C   INDICATE THE BEGINNING OF FRAME DEFINITION FOR THE SQUARE, AND
C   PROVIDE COMMANDS TO SCALE, ROTATE, AND TRANSLATE IT.
C
C   CALL UFRAME (FRAME(I))
C   CALL UMOVE X,X)
C   CALL UPSET ('ROTATE', X)
C   CALL USET ('RELATIVE')
C   CALL UPLYGN (0.,0.,4.,FLOAT(I))
C   CALL USET ('ABSOLUTE')
C
C   INDICATE THE TERMINATION OF EACH FRAME, AND CALL UNSHOW TO MAKE
C   THE NEWLY DEFINED FRAME INVISIBLE.
C
C   CALL UFREND (FRAME(I))
C   CALL UNSHOW (FRAME (I))
1  CONTINUE
C
C   AT THIS POINT, ALL OF THE FRAMES HAVE BEEN DEFINED, NOW WE WILL
C   DISPLAY EACH OF THE FRAMES IN SUCCESSION, FOR A TOTAL NUMBER OF
C   90 CYCLES = 10 OCCURRENCES OF EACH FRAME.
C
C   DO 2 I=1,90
C   J=MOD((I-1),9) + 1
C
C   CALL UNSHOW TO DISPLAY THE 'J'TH FRAME (MAKE IT VISIBLE), AND THEN
C   CALL UNSHOW TO MAKE IT INVISIBLE. SINCE THIS IS DONE AT RAPID RATE,
C   THE SQUARE WILL APPEAR TO BE DYNAMICALLY SCALED, ROTATED, AND
C   TRANSLATED.
C
C   CALL UNSHOW (FRAME(J))
C   CALL UNSHOW (FRAME(J))
2  CONTINUE
C   INDICATE END OF ALL GRAPHIC ACTIVITY AND TERMINATE PROGRAM.
C
C   CALL UEND
C   STOP
C   END

```

EXAMPLE XI-4



```

CHARACTER FRAME(8)
DATA FRAME/'A','B','C','D','E','F','G','H','I'/
CALL USTART
CALL UPSET ('LIBRARY',1.)
CALL UOUTLN
DO 1 I = 1, 8
X = 18.8 * FLOAT(I)
CALL UFRAME (FRAME(I))
CALL UMOVE (X,X)
CALL UPSET ('ROTATE',X)
CALL USET ('RELATIVE')
CALL UPLYEN (8.,8.,4.,FLOAT(I))
CALL USET ('ABSOLUTE')
CALL UFREND (FRAME(I))
CALL UGSHOW (FRAME(I))
1 CONTINUE
DO 2 I = 1, 88
J = MOD(I-1,8) + 1
CALL UGSHOW (FRAME(J))
CALL UGSHOW (FRAME(J))
2 CONTINUE
CALL UEND
STOP
END

```



**APPENDIX A**  
**ALPHABETICAL LISTING OF GCS SUBROUTINES**

UAIN	Accepts one character from the terminal CALL UAIN (ICHAR)
UALPHA	Insures that terminal is in alphanumeric mode CALL UALPHA
UAOUT	Outputs a character at current pen position subject to margining CALL UAOUT (ICHAR)
UAPEND	Adds GCS string terminator to character string CALL UAPEND (COUNT,DATAIN,DATOUT)
UARC	Draws an arc from current pen position CALL UARC (X,Y,ANGLE)
UASPCT	Forces the display dimensions to satisfy the specified aspect ratio CALL UASPCT (RATIO)
UAVERG	Fits a moving average curve to time series data CALL UAVERG (ARRAY,POINTS,FCST,PERIOD)
UAXIS	Draws axes with appropriate numeric and alphanumeric labeling CALL UAXIS (XMIN,XMAX,YMIN,YMAX)
UBAR	Draws a bar chart with appropriate numeric and alphameric labels CALL UBAR (ARRAY,PTS,LABELS,SIZE)
UBELL	Sounds the audible alarm at the terminal CALL UBELL
UCALL	Invokes a graphic data structure in two dimensions CALL UCALL (NAME,DX,DY,SX,SY,ANGLE)
UCHART	Draws a grouped bar chart for multi-valued data CALL UCHART (ARRAY,GROUPS,BARS,LABELS,YMAXL)
UCLOSE	Closes the current open frame/segment CALL CLOSE (SEGNAM)
UCONIC	Draws generalized conic sections

	CALL UCONIC (X,Y,P,E,THETA1,THETA2)
UCONTR	Draws contours on regular array of data CALL UCONTR (Z,X,Y,A,FX,FY,CURVE,FN)
UCOSYS	Creates a user coordinate plotting system CALL UCOSYS (DX,DY,SX,SY,ANGLE)
UCOUNT	Counts number of characters in character string CALL UCOUNT (DATA,COUNT)
UCRCLE	Draws a circle whose center location and radius are specified CALL UCRCLE (X,Y,RADIUS)
UDAREA	Sets the device display area associated with user window CALL UDAREA (XMIN,XMAX,YMIN,YMAX)
UDELET	Deletes a currently-defined frame/segment CALL UDELETE (SEGNAM)
UDIMEN	Adjusts physical boundaries of output device (alters aspect rates) CALL UDIMEN (XMAX,YMAX)
UDOIT	Perform various page layout functions CALL UDOIT (ACTION)
UDRAW	Draws solid line vector CALL UDRAW (X,Y)
UDRIN	Performs the input requested graphic operation and returns request CALL UDRIN (X,Y,ICHAR)
UEND	Terminates graphic operations and positions pen in home position CALL UEND
UERASE	Erases the screen or requests a clean plotting surface CALL UERASE
UERROR	Returns listing of source records with GCS error commentary CALL UERROR (ERLAST,TOTAL)
UFLUSH	Insures that visual display reflects all net program graphical output CALL UFLUSH

UFRAME	Defines the start of a named set of graphical commands CALL UFRAME (NAME)
UFREND	Defines the end of a named set of graphical commands CALL UFREND (NAME)
UGRIN	Gets coordinates and a character from terminal and returns them CALL UGRIN (X,Y,ICHAR)
UHISTO	Draws a histogram with appropriate numeric and alphanumeric labels CALL UHISTO (ARRAY,PTS,BARS)
UHOME	Moves beam to home position CALL UHOME
UIMAGE	Applies general 2-D image transformations to 'retained' segments CALL UIMAGE (X,Y,SX,SY,R,SEGNAM)
UINPUT	Inputs alphanumeric information from the current position CALL INPUT (DATA,COUNT,FLAG,OPTION)
UINVOK	Invokes a GCS structure at the current position CALL UINVOK (NAME)
ULINE	Draws a curve connecting two arrays of points in two spaces CALL ULINE (X,Y,PTS)
ULINFT	Calculates least squares linear fit to points provided CALL ULINFT (X,Y,XN,S,YI)
ULOOK	Establish portion display area onto which corresponding portion of current virtual space viewport will be mapped CALL ULOOK (XMIN,XMAX,YMIN,YMAX)
ULSTSQ	Calculates least squares polynomial fit to points provided CALL ULSTSQ (X,Y,XN,COEFF)
UMARGN	Sets the left and right, top and bottom alphanumeric window boundaries CALL UMARGN (XLEFT,XRIGHT,YBOTTM,YTOP)
UMENU	Menu board generating routine CALL UMENU (POINTS,LABELS,CHOICE)
UMODFY	Modifies setting of segment attributes



CALL UMODFY (SEGNAM,NAMAT,ATVALU)

UMOVE	Moves the pen to position specified by input arguments CALL UMOVE (X,Y)
UNSAVE	Restores all variables of the graphic status area CALL UNSAVE (ARRAY)
UNSHOW	Causes the named frame of graphical information to be made invisible CALL UNSHOW (NAME)
UNSVPN	Restores all pen related variables in the graphics status area CALL UNSVPN (ARRAY)
UNSVTR	Restores coordinate system related variables in the graphics status area CALL UNSVTR (ARRAY)
UOPEN	Opens a segment CALL UOPEN (SEGNAM,SEGTYP)
UORIGN	Creates a user coordinate system at the current beam/pen position CALL UORIGN
UOUTLN	Draws a box around the user's display area CALL UOUTLN
UPAUSE	Suspends execution until one character is entered from keyboard CALL UPAUSE
UPEN	Draws a line from current pen position to given coordinates CALL UPEN (X,Y)
UPEN1	Sets one 'USET' option for this call only before executing pen movement CALL UPEN1 (X,Y,OPTION)
UPIE	Draws a pie chart with appropriate numeric and alphameric labels CALL UPIE (ARRAY,PTS,LABELS,SIZE)
UPLACE	Applies 2-D translation image transformation to 'retained' segments CALL UPLACE (X,Y,SEGNAM)
UPLOT	General purpose multi-curve plotting routine CALL UPLOT (X,Y,CURVES,PARRAY,OPTION)

<b>UPLOT1</b>	Plots a single curve  CALL UPLOT1 (X,Y,PTS)
<b>UPLYGN</b>	Draws a regular polygon  CALL UPLYGN (X,Y,PTSIN,RADIUS)
<b>UPOINT</b>	Defines point which, together with two end points of a given line, defines the plane for the terminator and tic line  CALL UPOINT (X,Y,Z)
<b>UPOST</b>	Insures that only defined, visible segments are displayed  CALL UPOST
<b>UPRINT</b>	Prints information in hardware or software characters  CALL UPRINT (X,Y,INPUT)
<b>UPRNT1</b>	Allows alphanumeric output at current position with specified option  CALL UPRNT1 (DATA,OPTION)
<b>UPSET</b>	Changes setting in the gas which requires a parameter value to be set  CALL UPSET (OPTION,VALUE)
<b>UQUERY</b>	Obtains current value of specified variable in GSA  CALL UQUERY (OPTION,VALUE)
<b>UREAD</b>	Allows alphanumeric input from the graphic terminal  CALL UREAD (X,Y,DATA,COUNT,FLAG)
<b>URECT</b>	Draws a rectangle  CALL URECT (X,Y)
<b>URESET</b>	Resets GSA variables to default conditions  CALL URESET
<b>UROTAT</b>	Creates a user coordinate system at current position rotated as specified  CALL ROTAT (ANGLE)
<b>USAREA</b>	Changes device boundaries to maintain a one to one aspect ratio with the current window boundaries  CALL USAREA
<b>USAVE</b>	Saves all the variables of the Graphics Status Area  CALL USAVE (ARRAY)

<b>USAXIS</b>	Draws a single axis in any of three coordinates CALL USAXIS (AXIS,XSTART,YSTART,ZSTART,DIST)
<b>USCALE</b>	Creates a user coordinate system at current position with specified scale CALL USCALE (SX,SY)
<b>USCATR</b>	Draws a scatter plot CALL USCATR (X,Y,PTS)
<b>USET</b>	Sets a graphics status area variable to a given value CALL USET (OPTION)
<b>USHOW</b>	Causes the named frame of graphical information to be made visible CALL USHOW (NAME)
<b>USPLIN</b>	Fits a cubic spline curve to the input data CALL USPLIN (X,Y,PTS,RETX,RETY,RETPTS)
<b>USTART</b>	Initializes the graphics status area CALL USTART
<b>USTRCT</b>	Defines the start of a graphic data structure CALL USTRCT (NAME)
<b>USTUD</b>	Returns limits of virtual and display surfaces CALL USTUD(S)
<b>USVPN</b>	Saves all pen-related variables of the graphics status area CALL USVPN (ARRAY)
<b>USVTR</b>	Saves coordinate system related variables in the graphics status area CALL USVTR (ARRAY)
<b>UTAXIS</b>	Draws a time series axis with appropriate alphameric and numeric labels CALL UTAXIS (BEGIN,PERIOD,YMIN,YMAX)
<b>UTERM</b>	Defines the end of a graphic data structure CALL UTERM (NAME)
<b>UTILTY</b>	Performs data structure utility functions CALL UTILTY (OPTION,VALUE)
<b>UTSFIT</b>	Fits an exponentially smoothed curve to time series data



CALL UTSFIT (ARRAY,POINTS,FCST,ALPHA)

**UVIEW**            Defines position of viewer in relation to environment, and the direction of view.

CALL UVIEW (XVIEW,YVIEW,ZVIEW,XSITE,YSITE,ZSITE)

**UVWPLN**          Defines the location of the view (projection) plane

CALL UVWPLN (DISTAN)

**UVWPRT**          Defines the location of the view (projection) plane (obsolete, see UVWPLN)

CALL UVWPRT (DISTAN)

**UWAIT**            Waits a given number of seconds

CALL UWAIT (SECONDS)

**UWHERE**          Returns the coordinates of the current pen position in user units

CALL UWHERE (X,Y)

**UWINDO**          Sets the virtual window boundaries

CALL UWINDO (XMIN,XMAX,YMIN,YMAX)

**UWLOOK**          Adjusts both virtual window, and user display area to cover given portion of virtual space

CALL UWLOOK (XMIN,YMAX,YMIN,YMAX)

**UWRITE**          Prints information, then restores pen to location on input

CALL UWRITE (X,Y,DATA)

**UWRIT1**          Allows apphameric output at current position under one option

CALL UWRIT1 (DATA,OPTION)

**UZWNDO**          Sets the hither/yon window boundaries for Z-clipping

CALL UZWNDO (ZMIN,ZMAX)

**U3AXIS**          Creates a set of axes in three space

CALL U3AXIS (XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)

**U3CALL**          Invokes an existing graphics data structure in three space.

CALL U3CALL (X,Y,Z,SX,SY,SZ,RX,RY,RZ,NAME)

**U3CSYS**          Creates a new coordinate system in three space

CALL U3CSYS (X,Y,Z,SX,SY,SZ,RX,RY,RZ)

**U3DRAW**          Draws a solid line in three space

	CALL U3DRAW(X,Y,Z)
U3GRIN	Gets 3-D coordinates and a character from terminal and returns them CALL U3GRIN (X,Y,Z,ICHAR)
U3IMAG	Applies general 3-D image transformations to 'retained' segments CALL U3IMAG (X,Y,Z,SX,SY,SZ,RX,RY,RZ,SEGNAM)
U3LINE	Draws a curve connecting three arrays of points (X,Y,Z) in three spaces CALL U3LINE (X,Y,Z,PTS)
U3MOVE	Moves pen invisibly in three space CALL U3MOVE (X,Y,Z)
U3PEN	Draws a line from current pen position to given 3-D coordinates CALL U3PEN (X,Y,Z)
U3PEN1	Sets one 'USET' option for this call only before executing 3-D pen movement CALL U3PEN1(X,Y,Z,OPTION)
U3PLACE	Applies 3-D translation image transformation to 'retained' segments CALL U3PLAC(X,Y,Z,SEGNAM)
U3PLOT	Draws a general purpose graph in three space CALL U3PLOT (X,Y,Z,CURVES,PTS,OPTS)
U3PRNT	Displays textual data at pen position in three space CALL U3PRNT (X,Y,Z,DATA)
U3ROTA	Creates a user coordinate system in three space at current pen position, rotated as specified CALL U3ROTA (RX,RY,RZ)
U3SCAL	Creates a user coordinate system in three space at current pen position, scaled as specified CALL U3SCAL (SX,SY,SZ)
U3STUD	Gives current setting of the 3-D user display area and windows CALL U3STUD (ARRAY)
U3WHER	Returns current pen position in current units in three space CALL U3WHER (X,Y,Z)
U3WNDO	Sets the virtual 3-D window boundaries

CALL U3WDO (XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)

U3WRIT

Displays textual information in three space, returns pen to original position

CALL U3WRIT (X,Y,Z,DATA)



**APPENDIX B**  
**USET OPTIONS**

**TABLE 1**  
**USET OPTIONS MOST FREQUENTLY USED**

TO REQUEST ABSOLUTE COORDINATE PLOTTING (DEFAULT):

USET ('ABSOLUTE')	USET ('ABSO')
USET ('ABSb')	

TO REQUEST RELATIVE OR INCREMENTAL COORDINATE PLOTTING:

USET ('RELATIVE')	USET ('RELA')
-------------------	---------------

TO REQUEST RECTANGULAR COORDINATES (DEFAULT):

USET ('RECTANGULAR')	USET ('RECT')
----------------------	---------------

TO REQUEST POLAR COORDINATES:

USET ('POLAR')	USET ('POLA')
----------------	---------------

TO REQUEST ANGULAR VALUES IN DEGREES (DEFAULT):

USET ('DEGREES')	USET ('DEGR')
------------------	---------------

TO REQUEST ANGULAR VALUES IN RADIANS:

USET ('RADIANS')	USET ('RADI')
------------------	---------------

TO REQUEST PLOTTING IN VIRTUAL SPACE (DEFAULT):

USET ('VIRTUAL')	USET ('VIRT')
------------------	---------------

TO REQUEST PLOTTING IN DEVICE SPACE:

USET ('DEVICE')	USET ('DEVI')
-----------------	---------------

TO REQUEST DEVICE SPACE PLOTTING UNITS IN INCHES (DEFAULT):

USET ('INCHES')	USET ('INCH')
-----------------	---------------

TO REQUEST DEVICE SPACE PLOTTING IN RASTER UNITS:

USET ('RASTERUNITS')	USET ('RAST')
----------------------	---------------

TO REQUEST DEVICE SPACE PLOTTING IN FONT (CHARACTER SPACE) UNITS:

    USET ('FONTUNITS')      USET ('FONT')

TO REQUEST DEVICE SPACE PLOTTING IN PERCENT UNITS:

    USET ('PERCENTUNITS')      USET ('PERC')

TO REQUEST PLOTTING OF A VISIBLE LINE (DEFAULT):

    USET ('LINE')

TO REQUEST PLOTTING OF INVISIBLE LINES:

    USET ('MOVE')  
    USET ('NOLINE')      USET ('NOLI')  
    USET ('NOBLINE')      USET ('NOBL')

TO REQUEST PLOTTING OF LINES WITH ARROW TERMINATORS:

    USET ('ARROW')      USET ('ARRO')

TO REQUEST PLOTTING OF DASHED LINES:

    USET ('DASH')

TO REQUEST PLOTTING OF TIC LINES:

    USET ('TICLINE')      USET ('TICL')

TO REQUEST PLOTTING OF LINES WITH ONLY ENDPOINTS VISIBLE:

    USET ('POINT')      USET ('POIN')

TO REQUEST PLOTTING OF LINES COMPOSED OF CHARACTERS:

    USET ('ALPHANUMERICLINES')      USET ('ALPH')

TO REQUEST PLOTTING OF INVISIBLE LINES WITH CHARACTERS AT THEIR  
ENDPOINTS:

    USET ('CHARACTER')      USET ('CHAR')

TO REQUEST CHARACTER OUTPUT IN THE FORM OF HARDWARE CHARACTERS  
(DEFAULT):

    USET ('HARDWARE')      USET ('HARD')

TO REQUEST CHARACTER OUTPUT IN THE FORM OF SOFTWARE CHARACTERS:

    USET ('SOFTWARE')        USET ('SOFT')

TO REQUEST UPRINT/UWRITE OUTPUT IN TEXT FORMAT (DEFAULT):

    USET ('TEXT')

TO REQUEST UPRINT/UWRITE OUTPUT IN INTEGER FORMAT:

    USET ('INTEGER')        USET ('INTE')

TO REQUEST UPRINT/UWRITE OUTPUT IN REAL FORMAT:

    USET ('REALNUMBER')    USET ('REAL')

TO REQUEST PLOTTING WITH RESPECT TO THE SYSTEM AXIS (DEFAULT):

    USET ('SYSTEM')        USET ('SYST')

TO REQUEST PLOTTING WITH RESPECT TO THE USER AXIS:

    USET ('USERAXIS')      USET ('USER')

TO REQUEST A LINE TYPE/CHARACTER TERMINATOR COMBINATION:

    USET ('X\$bb')          USET ('X\$')

Where

    \$ IS THE DESIRED CHARACTER TERMINATOR

AND

    X, THE LINE TYPE SPECIFICATION, IS AS FOLLOWS:

A  ALPHANUMERIC LINE  
D  DASHED LINE  
L  SOLID LINE  
N  NULL OR INVISIBLE LINE  
T  TIC LINE  
b  denotes blank



TABLE 2

**USET OPTIONS BY ALPHABETICAL ORDER:**  
**(Default Options are in Bold Type)**

Option Name		To Request
'AARROW'		Alphanumeric lines with arrow terminators
'ABACKARROW'		Alphanumeric lines with back arrow terminators
'ABEND'	3D,*	To halt execution when error count exceeds certain limit
'ABORT'	3D	To halt execution when error count exceeds certain limit
'ABSb'		Plotting in an absolute coordinate system
'ABSOLUTE'		Plotting in an absolute coordinate system
'ABUTTING'		Abutting of display surface pages
'ACENTER'		To center character output about given location
'ACHARACTER'		Alphanumeric lines with character terminators
'ACOORDINATE'		Alphanumeric lines with ending coordinates indicated
'ADDITIVE'	*	Additive color blending mode
'ADOUBLEARROW'		Alphanumeric lines with arrowhead terminators
'ALLDISPLAYS'		Routing of graphical output to all devices
'ALPHANUMERIC'		Alphanumeric lines with no terminators
'ALTERNATEDISPLAY'		Routing of graphical output to first alternate device
'ANNUAL'		Time series axis scale in yearly intervals
'ANULL'		Alphanumeric lines with no terminators
'APOINT'		Alphanumeric lines with point terminators
'ARROWLINE'		Solid lines with arrow terminators
'ASYMBOL'		Alphanumeric lines with character terminators
'AUTOSCALE'		Automatic scaling for higher level graphing
'BACKARROWLINE'		Solid lines with back arrow terminators
'BALL'		Track ball graphical input
'BLACK'		Background color to be black
'BBLUE'		Background color to be blue
'BCYAN'		Background color to be cyan
'BESTFORMAT'		Numeric label output in best possible format
'BIHOURLY'		Time series axis scale in two hour intervals
'BLACK'		Switch to pen color black
'BLUE'		Switch to pen color blue
'BMAGENTA'		Background color to be magenta
'BRED'		Background color to be red
'BRIGHT'		Highest possible intensity for output
'BUILD'	3D	Structure building
'BWHITE'		Background color to be white
'BYELLOW'		Background color to be yellow
'CENTIMETERS'		Device space coordinates are in centimeters
'CHARACTER'		Null or invisible lines with character terminators
'CJUSTIFICATION'		Alphanumeric center justification
'COMPRESSED'	3D	Data structure editing option
'CONTINUOUS'		Curved lines be interpreted as one pen operation
'COORDINATES'		UPRINT/UWRITE output in (X,Y) coordinate format
'CURSOR'		Graphic cursor as graphic input device
'CWINDOWING'	3D	Circular windowing
'CYAN'		Switch to pen color cyan
'CYLINDRICAL'		Coordinates are to be of the form (R,THETA,Z).

NOTE: b - is a blank or space

		where R is the number of units of radius in the X,Y plane, THETA is the number of angular units around the Y axis, and Z is the number of units along the Z axis
'DAILY'		Time series axis scale in daily intervals
'DARROW'		Dashed lines with arrow terminators
'DASH'		Dashed lines with null terminations
'DATE'		Time series axis scale in date series interval
'DBACKARROW'		Dashed lines with arrow terminators
'DCHARACTER'		Dashed lines with character terminators
'DCOORDINATES'		Dashed lines with endpoint coordinates indicated
'DDOUBLEARROW'		Dashed lines with double arrow terminators
'DEFERRED'	*	Deferred error output until UEND is called
'DEGREES'		Angular information be interpreted in degrees
'DESENSITIZE'	I	Disabling of pick sensitivity
'DETECTABLE'		Enabling of pick sensitivity
'DEVICE'		Plotting in device space
'DIGITIZER'		Digitizer is graphics input device
'DIMb'		Lowest possible intensity for output
'DIMENSIONLINE'		Solid lines with arrow terminators
'DISPLAY'		Plotting in device space
'DNULL'		Dashed lines with no terminators
'DOUBLEARROW'		Solid lines with double arrow terminators
'DPOINT'		Dashed lines with point terminators
'DSYMBOL'		Dashed lines with character terminators
'DUMP'		To select dump option
'ECHO'	I	Echo alphanumeric input option
'EDGEAXIS'		X and Y axis labels at edge of graph
'ERROROUTPUT'		Immediate error output
'EXECUTE'	3D	Execution of data structure commands as they are built
'EXPANDED'	3D	Data structure editing option
'EXTENDEDMENU'	I,3D	Extended menuing option
'EXTRALARGE'		Extra large character size
'FAST'		Fast blink rate
'FITFRENCH'	*	Fit french curve to plotted points
'FITLINEAR'		Fit linear function to plotted lines
'FITPOLYNOMIAL'		Fit least squares polynomial to plotted points
'FITSPLINE'		Fit cubic spline curve to plotted points
'FNUMBERMODE'		Frame identifiers provided as numbers
'FONTUNITS'		To indicate device space plotting in font units
'FULLSCALE'		Full scaling for higher level graphing
'FUNCTIONKEYS'		Function key is graphics input device
'GAPPED'		Alternate light and dark line output
'GFORMAT'		Numeric label output in FORTRAN real (E or F) format
'GOTHIC'		Gothic character font
'GRADS'	3D	Angular units to be measured in grads
'GREEN'		Switch to pen color green
'GRIDAXIS'		Grid axes for higher level graphing
'HARDWAREFONT'		Output of hardware generated characters
'HIGHLIGHTED'		Highlighted segments
'HITHER/YONbCLIPPING'		Z axis clipping
'HORIZONTAL'	3D	Alphanumeric output to be printed horizontally
'HOURLY'		Time series axis scale in twenty-four hourly intervals
'IFORMAT'		Numeric label output in integer format
'IGNORE'	3D	Ignore duplicate studies on merge file

NOTE: b - is a blank or space

'INCHES'		Device space coordinates in inches
'INCREMENTAL'		Plotting in an incremental coordinate system
'INTEGER'		UPRINT/UWRITE output in integer format
'INVISIBLE'	3D	Invisible construction of frames and segments
'ITALICS'		Italic character format
'JOYSTICK'	I	Joystick as graphic input device
'KEYBOARD'	I	Keyboard as (pseudo) graphic input device
'LARGE'		Large character size
'LARROW'		Solid lines with arrow terminators
'LBACKARROW'		Solid lines with back arrow terminators
'LCHARACTER'		Solid lines with character terminators
'LCOORDINATES'		Solid lines with endpoint coordinates indicated
'LDOUBLEARROW'		Solid lines with double arrow terminators
'LEFT'	3D	Left handed coordinate system
'LETTER'		UPRINT/UWRITE output in text format
'LIGHTPEN'	I	Light pen as graphic input device
'LINE'		Solid lines with null terminators
'LINXAXIS'		Linear X axis drawing
'LINYAXIS'		Linear Y axis drawing
'LJUSTIFICATION'		Alphanumeric left justification
'LNULL'		Solid lines with null terminators
'LNXAXIS'		Natural log X axis drawing
'LNYAXIS'		Natural log Y axis drawing
'LOGARITHMIC'		Applies logarithmic transforms to all components
'LOGOBJECT'	3D	Logarithmic transforms are to be applied before any other transformations. (In this mode, log transforms may be applied to angle or radius components of 'CYLINDRICAL', 'POLAR', or 'SPHERICAL' coordinates)
'LOGORIGINALUNITS'		Application of log scaling before conversion to rectangular
'LOGSYSTEM'	3D	Logarithmic to be applied after conversion to 'SYSTEM' coordinates (in this mode, the logarithmic coordinate system axes.)
'LOGUSER'	3D	Logarithmic transforms are to be applied after conversion to 'ABSOLUTE', 'RECTANGULAR', 'USER' coordinates but before conversion to 'SYSTEM' coordinates. (In this mode, the logarithmic scaling will be applied along the current 'USER' coordinate system axes.)
'LOGXAXIS'		Base ten log X axis drawing
'LOGYAXIS'		Base ten log Y axis drawing
'LOWERCASE'		Lower case to be 'TEXT' case
'LPOINT'		Solid lines with point terminators
'LSYMBOL'		Solid lines with character terminators
'MAGENTA'		Switch to pen color magenta
'MEDIUM'		Medium character size
'MESSAGEDEVICE'		Alphanumeric I/O routed to a message device
'MILS'	3D	Angular units to be measured in mils
'MINUTELY'		Time series axis scale in minute intervals
'MONTHLY'		Time series axis scale in monthly intervals
'MOUSE'	I	Analog mouse as graphic input device
'MOVE'		Invisible lines with null terminators
'MULTIPLE'	3D	Multiple data structure invocation
'NARROW'		Invisible lines with arrow terminators
'NBACKARROW'		Invisible lines with back arrow terminators
'NCHARACTER'		Invisible lines with character terminators
'NCOORDINATES'		Invisible lines with double arrow terminators



'NDOUBLEARROW'		Invisible lines with double arrow terminators
'NEGATIVESIDE'	3D	Labels will be to the left or below the axes
'NEWSCALE'		New scale for higher level graphing
'NNULL'		Invisible lines with null terminators
'NOBLINE'		Invisible lines with null terminators
'NOABORT'	3D	Do not terminate if error count exceeds specified limit
'NOAXES'		No axes be drawn for higher level graphing
'NOBLINK'		No blinking to occur
'NOBUILD'	3D	No structure building
'NOCENTER'		Text output starts at given point
'NOCLIP'	3D	No device space clipping performed
'NODUMP'	3D	No dump performed
'NOECHO'	1,3D	No echoing of alphanumeric input
'NOEXECUTE''	3D	No execution of data structure commands as they are built
'NOFIT'		No curve fitting for higher level graphics
'NOHIGHLIGHTING'		No segment highlighting
'NOITALICS'		No slanting of software characters
'NOLINE'		Invisible lines with null terminators
'NOLOGARITHMS'	3D	To remove logarithmic transform application
'NONUNIFORM'	3D	High level plotting option
'NOMARK'		Invisible marks with null terminators
'NONABUTTING'		No abutting of display surface pages
'NONRETAINEDSEGMENTS'		Create segments in non-retained form
'NONUNIFORM'		Nonuniform scaling of higher level grading
'NOORIGIN'	3D	No origin to be forced for 'AUTOSCALE' or 'FULLSCALE' scaling options
'NOREPEAT'	3D	No coordinate repeating for high level plotting
'NOREWIND'		No rewind of structure save files
'NORMALINTENSITY'		Normal intensity for output
'NOSCRIPT'	3D	To disable any superscripting or subscripting of text output
'NOSIGNIFICANTZEROS'		Suppression of display of significant zeros
'NOSUPERSCRIPT'	3D	Same as 'NOSCRIPTING'
'NOTRAIL'	3D	Record of which GCS routines are invoked is not listed
'NOWINDOWING'	3D	Disable GCS windowing routine
'NOXLABEL'		No labels are to be drawn for graphing
'NOXREPEAT'	3D	To indicate that a component is provided for every X value in every curve in higher level graphing
'NOYLABEL'		No Y labeling for graphing
'NOYREPEAT'	3D	To indicate that a component is provided for every Y value in every curve in higher level graphing
'NOZCLIPPING'		No hither/yon clipping
'NOZLABELS'	3D	To indicate that no Z labels are to be drawn for higher level graphing
'NOZREPEAT'	3D	To indicate that a component is provided for every Z value in every curve in higher level graphing
'NOBLINE'		Invisible line with null terminator
'NOBMARK'		Line type
'NPOINT'		Invisible lines with point terminators
'NSYMBOL'		Invisible lines with character terminators
'OLDSCALE'		Old scale to be used for higher level graphing
'ORIGIN'	3D	An origin to be forced for 'AUTOSCALE' and 'FULLSCALE' scaling options
'ORMODE'	1,3D	Asynchronous event processing option
'ORTHOGRAPHIC'	3D	To specify orthographic projection in which the

NOTE: b - is a blank or space

'OWNSCALE'		projection is parallel from all points
'PARALLELLABELS'	3D	Own scale option for higher level graphing To specify that the main axis of the numeric labels will be parallel to the axis
'PENAXIS'		Axis intersection at current position for graphing
'PENDOWN'		Solid lines with null terminators
'PENORIGIN'	3D	To force the current pen position to be included in the axis range.
'PENUP'		Invisible lines with null terminators
'PERIODIC'		Time series axis scale in accounting period intervals
'PERCENTUNITS'		Device space coordinates specified in percent units
'PERPENDICULARLABELS'	3D	To specify that the major axis the numeric labels will be perpendicular to the axis
'PERSPECTIVE'	3D	To specify perspective projection in which line length diminishes as the distances from the viewing position become greater
'PIRADIANS'	3D	Angular information be interpreted in PI radians
'PLAINAXIS'		Plain axes to be drawn for high level graphing
'PLOTDEVICE'		Alphanumeric I/O to the plotting device
'POINT'		Invisible lines with point terminators
'POLAR'		Plotting in polar (RHO, THETA) units
'POSITIVESIDE'	3D	Labels will be above or to the right of the axis
'PRIMARYDEVICE'		Routing of graphical output to primary device
'QUARTERLY'		Time series axis scale in quarter year intervals
'RADIANS'		Angular information be interpreted in radians
'RASTERUNITS'		To indicate device space coordinates are specified as is in raster units
'REAL'		UPRINT/UWRITE output in real number format
'RECTANGULAR'		Plotting on the user's reference axis
'REDb'		Switch to pen color red
'REFERENCE'		Plotting on the user's reference axis
'REFRESHEDSEGMENT'		Segments to be retained
'RELb'		Plotting in a relative coordinate system
'RELATIVE'		Plotting in a relative coordinate system
'REPLACE'	3D	Data structure building option
'RETAINEDSEGMENTS'		Segments to be retained structures from merge file
'REWIND'	3D	Data structure file handling command
'RIGHTHAND'	3D	Right handed coordinate system
'RJUSTIFICATION'		Alphanumeric right justification
'RWINDOWING'	3D	Rectangular windowing
'SECONDLY'		Time series axis scale in second intervals
'SECRET'		Security classification secret
'SEGMENTED'		Curved lines be interpreted as multiple pen operations
'SEMIANNUAL'		Time series axis scale in semi annual intervals
'SENSITIZE'	3D	Make graphic segments visible
'SIGNIFICATZEROES'		Display of significant zero
'SIMULATED HARDWARE CHARACTERS'		Output of simulated hardware characters
'SINGLE'	3D	Data structure invocation option
'SITEPOINT'	3D	Viewpoint distance to be measured from the view site
'SLOWBLINK'		Slow blink rate
'SMALL'		Use smallest hardware character size
'SOFTWAREFONT'		Output of software generated characters

NOTE: b - is a blank or space

'SONICPEN'		Sonic pen is graphics input device
'SPECIFIC'		To specify particular device units instead of percent units
'SPHERICAL'		Coordinates are of the form (R,THETA,PHI) where R is the number of units of radius, THETA is the number of angular units around the Z axis, and PHI is the number of angular units around the X axis
'STANDARDMENU'	1,3D	Menuing option
'SUBSCRIPT'	3D	To specify that the output will be lowered from the specified line of text
'SUPERScript'	3D	To specify that the output will be raised from the specified line of text.
'SUPPRESSERRORS'		Error output be suppressed
'SYMBOL'		Invisible lines with character terminators
'SYSTEMAXIS'		Plotting on the system axis
'TABLET'	1	Analog tablet as graphic input device
'TARROW'		Tic lines with arrow terminators
'TBACKARROW'		Tic lines with back arrow terminators
'TCHARACTER'		Tic lines with character terminators
'TCOORDINATES'		Tic lines with endpoint coordinates indicated
'TDOUBLEARROW'		Tic lines with double arrow terminators
'TEXT'		UPRINT/UWRITE output in text format
'SUBTRACTIVE'		Subtractive color blending mode
'TICAXES'		Tic axes to be drawn for higher level graphing
'TICLINE'		Tic lines with null terminators
'TNULL'		Tic lines with null terminators
'THINLINES'		Line width to be thin
'TOPSECRET'		Security Classification Top Secret
'TPOINT'		Tic lines with point terminators
'TRAIL'	3D	To indicate by an identification number which GCS routine is involved.
'TSYMBOL'		Tic lines with character terminators
'TWELVEHOUR'		Time series axis scale in twelve hour intervals
'TWENTYFOURHOUR'		Time series axis scale in twenty four-hour intervals
'UNCLASSIFIED'		Security classification unclassified
'UNDETECTABLE'		Disabling pick sensitivity
'UNIFORM'	3D	High level graphing system
'UNINTERRUPTED'		Non-gapped line output
'UPPERCASE'		Upper case to be 'TEXT' case
'USER'		Plotting on a user defined axis system
'VERTICAL'	3D	Alphanumeric output to be spaced vertically
'VIEWPOINT'	3D	View port distance to be measured from the view point
'VIRTUAL'		Plotting in virtual space
'VISIBLE'	3D	Visible framed output
'WEEKLY'		Time series axis scale in weekly intervals
'WHITE'		Switch to pen color white
'WIDELINES'		Line width to be wide
'WORKING'		New cumulative user coordinate system
'WORLDCOORDINATESYSTEM'		Coordinate system to be the default axis system
'XABSOLUTE'	3D	To specify the X coordinates with respect to the origin of the current coordinate system for indicated components. Y and Z components are to be specified with respect to the current beam/pen position
'XALPHANUMERIC'		An X axis alphanumeric label
'XAXIS'		The X axis be drawn for high level graphing

NOTE: b - is a blank or space



'XBOTHLABELS'		X axis alphanumeric and numeric labels
'XCONSTANT'	3D	To indicate that the X component does not vary during the drawing of any curve in higher level graphics
'XEDGEZEROAXIS'		The X axis at edge of graph
'XLOGARITHMIC'		Logarithmic X and linear Y plotting
'XNEGATIVE'	3D	Negative X axis represents up in 3D graphics
'XNUMERIC'	3D	An X axis numeric label
'XPOSITIVE'	3D	Positive X axis represents up in 3D graphics
'XRELATIVE'	3D	To specify the X coordinates with respect to the current beam/pen position. Y and Z components are to be specified with respect to the origin of the current coordinate system
'XREPEAT'	3D	To indicate that one set of X values is provided which will be reused for every curve in higher level graphing
'XYAXES'		The X and Y axes be drawn for high level graphing
'XYABSOLUTE'	3D	To specify the X and Y coordinates with respect to the origin of the current coordinate system for the indicated components. Z components are to be specified with respect to the current beam/pen position
'XYCOORDINATES'		To specify that the data printed by UPRINT/UWRITE is in the form of an (X,Y) coordinate pair.
'XYLOGARITHMIC'	3D	Logarithmic X and Y plotting, linear Z plotting
'XYPLANE'	3D	Labels are to be drawn in the plane formed by the X and Y axes
'XYRELATIVE'	3D	To specify the X and Y components with respect to the current beam/pen position. Z components are to be specified with respect to the origin of the current coordinate system
'XYVIEW'	3D	To view plane formed by X and Y axes
'XYZAXES'	3D	All three axes are to be drawn for higher level graphing
'XYZCOORDINATES'	3D	Text printed by U3PRNT/U3WRIT to be in form of (X,Y,Z) triplet
'XYZLOG'	3D	Applies logarithmic transforms to all three components
'XYZVIEW'	3D	To view all three axes
'XYZb'	3D	To set the rotation application order as indicated
'XYCOORDINATES'		UPRINT/UWRITE output in (X,Y) coordinate format
'XZABSOLUTE'	3D	To specify the X and Z coordinates with respect to the origin of the current coordinate system for the indicated components. Y components are to be specified with respect to the current beam position
'XZAXES'	3D	The X and Z axes are to be drawn for higher level graphing
'XZEROYEDGEAXIS'		The X axis adjacent to boundary of display area
'XZLOGARITHMIC'	3D	Applies Logarithmic transformation to X and Z components
'XZPLANE'	3D	Label plane to be plane formed by X and Z axis
'XZRELATIVE'	3D	To specify the X and Z components with respect to the current beam/pen position. Z components are to be specified with respect to the origin of the current coordinate system
'XZVIEW'	3D	To view the plane formed by the X and Z axes
'XZYb'	3D	To set the rotation application order as indicated

NOTE: b - is a blank or space

'YABSOLUTE'	3D	Y coordinates are specified with respect to the origin of the current coordinate system for the indicated units. X and Z components are specified with respect to the current beam/pen position
'YALPHANUMERIC'		Y axis alphabetic label
'YAXIS'		The Y axis to be drawn for high level graphing
'YBOTHLABELS'		Y axis having alphabetic and numeric labels
'YCONSTANT'	3D	To indicate that the Y component does not vary during the drawing of any curve
'YEARLY'		Time series axis scale in yearly intervals
'YEDGEZEROAXIS'		The Y axis at edge of graph
'YELLOW'		Switch to pen color yellow
'YLOGARITHMIC'		Logarithmic Y plotting
'YNEGATIVE'	3D	Negative Y direction represents up in 3D graphics
'YNUMERIC'		Y axis numeric label
'YPOSITIVE'	3D	Positive Y direction represents up in 3D graphics
'YRELATIVE'	3D	Y coordinates are specified with respect to the current beam/pen position for the indicated components X and Z components are to be specified with respect to the origin of the current coordinate system
'YREPEAT'	3D	To indicate that one set of Y values is provided which will be reused for every curve in higher level graphing
'YXZb'	3D	To set the rotation application order as indicated
'YZABSOLUTE'	3D	Y and Z coordinates are specified with respect to the origin of the current coordinate system for the indicated under X components are specified with respect to the current beam/pen position
'YZAXES'	3D	The Y and Z axes to be drawn for higher level graphing
'YZPLANE'	3D	Label plane to be plane formed by Y and Z axes
'YZERXEDGEAXIS'		The Y axis adjacent to boundary of display area
'YZLOGARITHMIC'	3D	Applies logarithmic transformations to Y and Z components
'YZRELATIVE'	3D	Y and Z coordinates to be specified with respect to the current beam/pen position for the Y and Z components, and the X component to be specified with respect to the current beam/pen position for the Y and Z components, and the X component to be specified with respect to the origin of the current coordinate system
'YZVIEW'	3D	To view the plane formed by the Y and Z axes
'YZXb'	3D	To set the rotation application order as specified
'ZABSOLUTE'	3D	Z coordinates to be specified with respect to the origin of the current coordinate system for the indicated component. X and Y components are to be specified with respect to the current beam/pen position
'ZALPHANUMERIC'	3D	Alphanumeric labels to be drawn for higher level drawing
'ZAXIS'	3D	Z axis is to be drawn for higher level graphing
'ZBOTHLABELS'	3D	Both numeric and alphanumeric labels to be drawn for higher level drawing
'ZCLIP'	3D	Clip in Z direction
'ZCONSTANT'	3D	To indicate that the Z component does not vary during the drawing of any curve
'ZEROAXES'		X and Y axes adjacent to boundary of display area

NOTE: b - is a blank or space

<b>'ZLOGARITHMIC'</b>	3D	Applies logarithmic transform to Z component
<b>'ZNEGATIVE'</b>	3D	Negative Z axis represents up direction in 3D graphics
<b>'ZNUMERIC'</b>	3D	Numeric Z labels to be drawn for high level graphing
<b>'ZPOSITIVE'</b>	3D	Positive Z axis represents up direction in 3D graphics
<b>'ZRELATIVE'</b>	3D	Z coordinates are to be specified with respect to the current beam/pen position. X and Y components are to be specified with respect to the origin of the current coordinate system
<b>'ZREPEAT'</b>	3D	To indicate that one set of Z values is provided which will be reused for every curve in higher level graphing
<b>'ZXYb'</b>	3D	To set the rotation application order as indicated
<b>'ZYGb'</b>	3D	To set the rotation application order as indicated
<b>'12HOUR'</b>		Twelve hour time axis
<b>'13WEEK'</b>		Thirteen week time axis
<b>'2DCOORDINATES'</b>	3D	To specify A coordinate terminator in which two components are listed. (This option is independent of the text coordinate options of 'XYCOORDINATES' and 'XYZCOORDINATES')
<b>'24HOUR'</b>		Twenty four time axis
<b>'3DCOORDINATE'</b>	3D	To specify a coordinate terminator in which all three components are listed. (This option is independent of the text coordinate option of 'XYCOORDINATES' and 'XYZCOORDINATES')



## APPENDIX C

### UPSET OPTIONS

Option Name	Value
'ANGLE OF TEXT'	Is an angular value which specifies the angle of the text string in relation of the current X axis. Default value is 0.
'ATTENTION QUEUE SIZE'*	An integer indicating the number of words provided in the attention queue. Default is 0.
'ATTITUDE'	Is an angular value which specifies the orientation of the up direction axis with the sides of the window. Default value is 0. Meaning that the up direction is parallel with the left and right window boundaries and pointing towards the top of the window.
'BACKGROUND COLOR'	Is an integer which specifies the color index within the color table for colored backgrounds. Values 0-7 are predefined to represent black, white, red, green, yellow, blue, magenta, and cyan, respectively. Default color is black or none.
'BASE OF LOGARITHMS'	Is a positive value which specifies the base of the logarithms used to perform logarithmic scaling. Default base is 10. This option sets the specified base along each coordinate component.
'BRIGHTNESS'	Is a value between 0 and 100. percent indicating the position in the range of possible line intensity settings from dimmest to brightest. Default brightness is 60%.
'CHARACTER'	Is a Hollerith character which becomes the current system character. The default system character is '*'.
'COLOR'	Is an integer which specifies the color index within the color table. Values 0-7 are predefined to represent black, white, red, green, yellow, blue, magenta, and cyan, respectively. Default color is device-dependent.
'COPY DELAY'	Is a value which indicates the number of seconds of delay required during generation of a hard copy. Default value is device-dependent and is preset to the value required by the selected device.
'DESCRIPTOR FILE'*	Is a Fortran file number representing the file containing the font descriptors. Default is zero indicating no font file specified.

\*means not implemented

'DISTANCE'	Is a value measured from the current view plane distance base specifying the position of the view (projection) plane. The default is 0. measured from the view site.
'FONT NAME'*	Is a Hollerith string indicating the desired character font. Default is 'GCS' which is the most efficient font.
'GREYSCALE'	Is a value indicating a particular grey level for terminals which support multiple grey scales rather than colors.
'GRID SPECIFICATION'	Is a value containing a dash specification to be used when generating grid axes. Default is 0. which indicates a solid line should be used.
'HORIZONTAL SIZE'	Indicates the width of a software character position in current user units. Default is 5 virtual units.
'INPUT FILE'	Is a Fortran file number indicating which file will be used to obtain graphics input. The default is set to the appropriate computer-system dependent file.
'LABELbROTATION'	is the number of angular units the axes lbaels are to be rotated around the axes.
'LIBRARY FILE'	Is a Fortran file number indicating which file should be used by the GCS structure and segmentation facilities as a random work file. Default is 0. indicating no file has been provided.
'LOWER'	Is a Hollerith character which will be used by GCS as the indication to shift to lower case. Default character is '>'.
'MARKER INDEX'	Is an integer value selecting a marker/symbol. Default marker symbol is 0. indicating a point.
'ORIENTATION'	Is an angular value indicating the display orientation of GCS created software symbols and figures such as software characters, polygons, and rectangles. Default orientation is 0.
'OUTPUT FILE'	Is a Fortran file number indicating which file will be used for sending graphics output to the display device. The default is set to the appropriate computer system dependent file.
'POLYNOMIAL DEGREE'	Specifies the degree of the polynomial to be created in calculating a least squares fit through a collection of points. Default value is 5.
'PRECISION'	Specifies the number of significant digits to
*means not implemented	

	appear when displaying real numbers. Default value is 4.
'READFILE'	Is a Fortran file number indicating which file will be used to obtain non-graphic input. The default is set to the appropriate computer system dependent file
'ROTATION'	Same as 'ORIENTATION'.
'SCALEFACTOR'	Specifies a scale to be applied to GCS-created geometric figures such as polygons and rectangles.
'SCRIPTLEVEL'	Is an integer value indicating the scripting level to be set for textual output. Default value is 1.
'SETDASH'	Specifies the characteristics of the dashed lines to be plotted by UPEN. Default value is 56.
'SIZE'	Is a positive integer value which sets hardware character sizes. Values of 1 through 4 correspond to USET options 'SMALL', 'MEDIUM', 'LARGE', and 'EXTRA LARGE' respectively. Default value is 1 for 'SMALL' characters.
'SLANTANGLE'	Is an angular value indicating the amount of slant from the vertical for italicized software characters. Default value is approximately 18 degrees.
'SPAN ANGLE'	Is an angular value indicating the portion of a circle to be occupied by the pie chart. Default value is 360. degrees.
'SPECIFICATION UNITS'	Is a positive value indicating the number of specification units contained in device space for all directions. Default value is 1000.
'SPEED'	Specifies the speed of the communication line in characters per second. Default value is system dependent.
'START ANGLE'	Is an angular value indicating the starting position of the first wedge of the pie chart. Default value is 0.
'STRUCTURE TABLE SIZE'	Is an integer value indicating the number of words in the user provided structure table. Default value is 100.
'SUBSCRIPT CHARACTER'	Is a Hollerith character which will be used to decrease the scripting level by 1. Default subscript character is ' '.
'SUPERScript CHARACTER'	Is a Hollerith character which will be used to increase the scripting level by 1. Default superscript character is ' '.

\*means not implemented



'SZMARKER'	Is a value in current device units which specifies the size of software generated markers. Default value is device-dependent.
'TABHORIZONTAL'	Is an array of 10 elements containing 10 tab positions in current device units. Default value has all tab stops set to zero.
'TABVERTICAL'	Is an array of 10 elements containing 10 vertical tab positions in current device units. Default value has all tab stops set to zero.
'TERMINATOR'	Is a Hollerith character which will be used as the GCS string terminator character. Default value is ' '.
'TICINTERVAL' → 'TICLENGTH'	Specifies the distance in current user units between tic marks of a UPEN created tic line. Default value is 10.
'TICMINUS'	Specifies in current units the size of that portion of a tic mark which lies on the clockwise side of the tic line. Default value is .05 inches.
'TICPLUS'	Specifies in current units the size of that portion of a tic mark which lies on the clockwise side of the tic line. Default value is .05 inches.
'TICX'	Specifies the distance between tic marks or grid lines along X axes. Default value is 0, indicating that a 'nice' number should be chosen.
'TICY'	Is the same as 'TICX' for Y axes.
'TICZ'	Is the same as 'TICX' for Z axes.
'UPPER'	Is a Hollerith character which will be used by GCS as the indication to shift to upper case. Default character is '<'.
'VERTICAL SIZE'	Indicate the height of a software character position in current user units. Default value is seven virtual units.
'WIDTH'	Is a value in current units of the width of a line. Default value is 0, indicating a thin line.
'WRITE FILE'	Is a Fortran file number indicating which file will be used to generate non-graphic output. The default is set to the appropriate computer system dependent file.
'XBASE OF LOGS'	Is a positive value which specifies the base for logarithmic scaling along the X component. Default value is 10.
'XLABEL'	Specifies the alphanumeric label to be displayed along X axes. Default value is 'X'.

\*means not implemented

AD-A063 167

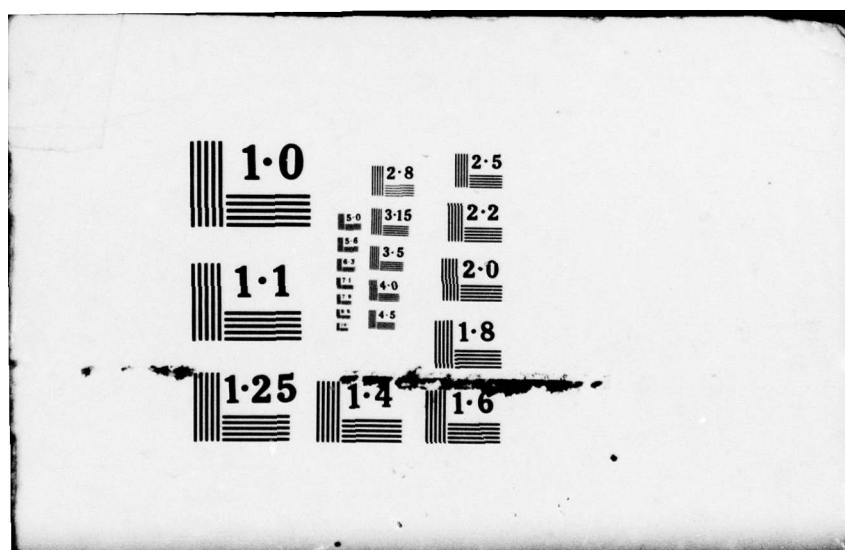
ARMY ENGINEER WATERWAYS EXPERIMENT STATION VICKSBURG MISS, F/G 9/2  
GRAPHICS COMPATIBILITY SYSTEM (GCS). PRIMER ON COMPUTER GRAPHIC--ETC(U)  
1978

UNCLASSIFIED

NL

3 OF 5  
AD A  
083167







'XPERCENT'	Is a value specifying the portion of the width of a software character position to be occupied by a character. Default value is .65 indicating 65% of the width.
'XROTATION'	Is an angular value indicating the amount of rotation around the X axis for UNIVOK structure invocations. Default value is 0.
'XSCALE'	Is the scale factor to be applied along the X axis for UNIVOK structure invocations. Default value is 1.
'XSIZE'	Is the size of hardware or simulated hardware character positions in current device units. Default value is device-dependent and corresponds to 'SMALL' hardware character size.
'XSPECIFICATION UNITS'	Is a positive value indicating the number of X specification units in device space. Default value is 1000.
'YBASE OF LOGS'	Is a positive value indicating the number of X specification units in device space. Default value is 1000.
'YBASE OF LOGS'	Is a positive value which specifies the base for logarithmic scaling along the Y component. Default value is 10.
'YLABEL'	Specifies the alphanumeric label to be displayed along the Y axis. Default value is 'Y'.
'YPERCENT'	Is a value specifying the portion of the height of a software character position to be occupied by a character. Default value is .65 indicating 65% of the height.
'YROTATION'	Is an angular value indicating the amount of rotation around the Y axis for UNIVOK structure invocations. Default value is 0.
'YSCALE'	Is the scale factor to be applied along the Y axis for UNIVOK structure invocations. Default value is 1.
'YSIZE'	Is the size of hardware or simulated hardware character positions in current device units. Default value is device-dependent and corresponds to 'SMALL' hardware character size.
'YSPECIFICATION UNITS'	Is a positive value indicating the number of Y specification units in device space. Default value is 1000.
'ZBASE OF LOGS'	Is a positive value which specifies the base for logarithmic scaling along the Z component. Default value is 10.

\*means not implemented

**'ZLABEL'**

Specifies the alphanumeric label to be displayed along the Z axis. Default value is 'Z'.

**'ZROTATION'**

Is an angular value indicating the amount of rotation around the Z axis for UINVOK structure invocations. Default value is 0.

**'ZSCALE'**

Is the scale factor to be applied along the Z axis for UINVOK structure invocations. Default value is 1.

**\*means not implemented**

## APPENDIX D

### GCS DEFAULT CONDITIONS

This appendix addresses those options which are present in the Graphics Status Area as default options. After a call to Subroutine USTART, the Graphics Compatibility System is set to the default conditions, as indicated. The default options can be divided into two groups: Basic Plotting Options and High Level Plotting Options.

#### I. Default Basic Plotting Options

Plotting is done in 'RECTANGULAR' and 'ABSOLUTE' coordinates on the 'SYSTEM' coordinate axis. The 'USER' coordinate axes are identical to the system axis. Plotting is done in 'VIRTUAL' space with a virtual window whose limits are from 0.0 to 100.0 in the X direction, and from 0.0 to 100.0 in the Y direction. The virtual window is mapped into a display area which is the largest square area on the device display surface. The right hand edge of the square corresponds to the right edge of the display surface, and the top edge of the square corresponds to the top edge of the display surface.

Character output in GCS will be, by default, in 'HARDWARE' character format, of type 'GOTHIC' and of 'MEDIUM' size. If 'SOFTWARE' characters are requested via USET then the default horizontal size is five (5.0) virtual units, and the default vertical size is seven (7.0) virtual units.

By default, angular units for angular specifications are in 'DEGREES'. If the user switches to 'DEVICE' space, then all length or distance units (i.e. rectangular coordinates) are in terms of 'INCHES'. The default alphanumeric margins are the boundaries of the plotting surface. For alphanumeric I/O through GCS (i.e. from UPRINT or UREAD), the data will be assumed to be in 'TEXT' mode. The alphanumeric output defaults to the 'PLOT' device if possible, and any graphic output will go to all devices in a cluster. Graphic input is from the primary input device; the number of digits of precision for numeric output is four (4); and GCS detected errors are signalled as they occur.

A line which is drawn by a subroutine in GCS is considered to consist of two parts; a line type and a line terminator. The line type may be solid (visible), ticked, null (invisible), dashed, or alphanumeric. The line may be terminated by an arrow, a back arrow, double arrows, a character, a point, a symbol, a set of coordinate values, or nothing at all. The default line type in GCS is 'SOLID' with 'NULL' terminators ('LNULL'). If 'TICLINES' are requested via USET option, then the default tic interval is ten (10.0) virtual units. It is the user's responsibility to insure that the tic interval is appropriate if he switches to 'DEVICE' space or alters the default virtual window setting or the default display area setting. If 'DASHLINES' are requested, then the default dash specification (56.) will result in a dashed line which is alternately light and dark, in increments of approximately 0.075 inches. If a 'CHARACTER' line terminator is requested but no character is specified by way of Subroutine UPSET, then the character asterisk (\*) is used. Similarly, the asterisk is used to compose the line type if 'ALPHANUMERIC' lines are requested.

For 3D operations, the viewing environment is set up to simulate a 2D only environment. The view point is located at (0.,0.,150.), the view site is at (0.,0.,0.) and the view plane is located at the view site. The system coordinate system is considered to be 'RIGHT-HANDED' with the 'ZPOSITIVEAXIS' representing up. Note that since the view is down the Z axis, the up direction degenerates to 'YPOSITIVE'. The Z axis clipping planes are 150. (hither plane) and 1.0E+30 (yon plane) with 'NOZCLIPPING'.



## II. Default High Level Plotting Options

For each call to UPLLOT or UPLLOT1, the data values which represent the curves are examined, and a 'NEWSCALE' is created. UAXIS will be invoked to create 'XYAXES'. The data values will be examined, appropriate limits for the established. Numeric labels only will be output for the X axis and the Y axis. The axes will be positioned at the 'EDGE' of the plot. Both the X axis and the Y axis will be drawn in a linear coordinate space. The axis lines will be ticked. If a time series axis is plotted by invoking Subroutine UTAXIS, the default interval for the X axis will be 'DAILY'. No curves will be fit to the data values, but if 'FITPOLYNOMIAL' is requested, then subroutine UPLLOT will attempt to fit a fifth degree polynomial to the data.

## III. Summary

COORDINATE SPACE:	'VIRTUAL'
COORDINATE TYPE:	'ABSOLUTE'
COORDINATE SYSTEM:	'RECTANGULAR'
COORDINATE AXIS:	'SYSTEM'
VIRTUAL WINDOW:	0.0 TO 100.0 X DIRECTION 0.0 TO 100.0 Y DIRECTION
DISPLAY AREA:	Largest square area which is right-up on the display surface of the device.
DEVICE SPACE UNITS:	'INCHES'
ANGULAR UNITS:	'DEGREES'
LINE TYPE:	'LINE' or 'LNULL'
SYSTEM CHARACTER:	asterisk (*)
TIC INTERVAL:	10.0 virtual units
DASH SPECIFICATION:	56.
CHARACTER TYPE:	'HARDWARE'
CHARACTER FONT:	'GOTHIC'
CHARACTER SIZE:	'MEDIUM'
SOFTWARE CHARACTER SIZE:	5.0 virtual units horizontal 7.0 virtual units vertical
INPUT/OUTPUT FORMAT:	'TEXT'
ALPHANUMERIC MARGINS:	Device display surface boundaries.
OUTPUT ROUTE:	'PLOTDEVICE'
OUTPUT DISTRIBUTION:	'ALLDEVICES'
GRAPHIC INPUT:	Primary input device
DIGITS OF PRECISION:	4
ERROR HANDLING:	'IMMEDIATE OUTPUT'
AXIS SCALING:	'AUTOSCALE'
AXIS LABELING:	'XNUMERICLABEL' 'YNUMERICLABEL'
AXIS POSITIONING:	'EDGEAXIS'
AXIS TYPE:	'TICAXIS'
AXIS EXISTENCE:	'XYAXES'
AXIS COORDINATES:	'LINXAXIS' 'LINYAXIS'
TIME SERIES AXIS SCALE:	'DAILY'

## APPENDIX E

### USSET OPTIONS BY CLASS

#### Coordinate Type

'ABSOLUTE'	'INCREMENTAL'	'RIGHTHANDED'	'ZABSOLUTE'
'RELATIVE'	'XYRELATIVE'	'YABSOLUTE'	'ZRELATIVE'
'XABSOLUTE'	'XZABSOLUTE'	'YRELATIVE'	
'XRELATIVE'	'XZRELATIVE'	'YZABSOLUTE'	
'XYABSOLUTE'	'LEFTHANDED'	'YZRELATIVE'	

#### Coordinate Type

'RECTANGULAR'	'CYLINDRICAL'	'LOGOBJECT'	'XYZLOGARITHMIC'
'POLAR'	'SPHERICAL'	'LOGSYSTEM'	'XZLOGARITHMIC'
'LOGARITHMIC'	'XLOGARITHMIC'	'LOGUSER'	'YZLOGARITHMIC'
'YLOGARITHMIC'	'XYLOGARITHMIC'	'NOLOGARITHMS'	'ZLOGARITHMIC'

#### Coordinate Space

'VIRTUAL'	'SPECIFIC'
'DEVICE'	'DISPLAY'

#### Device Space Units

'INCHES'  
'CENTIMETERS'  
'FONTUNITS'  
'PERCENTUNITS'  
'RASTERUNITS'

#### Angular Units

'DEGREES'	'GRADS'
'RADIANs'	'MILS'
'PIRADIANs'	

#### Frame Composition

'INVISIBLE'  
'VISIBLE'

#### Line Type

'LINE'	'LNULL'	'NOLINE'	'LBACKARROW'
'DASH'	'DNULL'	'NOMARK'	'LCOORDINATE'
'POINT'	'NPOI'	'NNULL'	'NO LINE'
'TICLINE'	'TNULL'	'LARROW'	'NO MARK'
'CHARACTER'	'BACKARROWLINE'	'LDOUBLEARROW'	'DIMENSIONLINE'
'SYMBOL'	'COORDINATELINE'	'MOVE'	'NSYMBOL'
'ALPHANUMERIC'	'NCHA'	'ARROWLINE'	'ANULL'
'DOUBLEARROWLINE'			

Lines that are drawn by GCS are composed on a line type and a line terminator. A large number of line types and terminators are possible. The specification is composed by

combining the first letter of the line type with the name of the terminator. The following tables illustrate the possible linetypes and terminators:

LINE TYPE	FIRST LETTER
-----------	--------------

LINE	L
DASH	D
ALPHANUMERIC	A
NULL (INVISIBLE)	N
TICLINE	T

**Line Terminators**

NULL (NO TERMINATORS)  
 CHARACTER  
 SYMBOL  
 COORDINATE  
 POINT  
 ARROW  
 BACKARROW  
 DOUBLEARROW

**Line Repeatability**

'UNINTERRUPTED'  
 'GAPPED'

**Curve Approximation**

'CONTINUOUS'  
 'SEGMENTED'

**Blink Rate**

'NOBLINK'  
 'FASTBLINK'  
 'SLOWBLINK'

**Color**

'WHITE'	'BLUE'
'BLACK'	'RED'
'GREEN'	'MAGENTA'
'CYAN'	'YELLOW'

**Intensity**

'DIM'  
 'BRIGHT'

**Coordinate Axis**

'SYSTEMAXIS'  
 'USERAXIS'

**Coordinate Axis Composition**

'WORKINGAXIS'  
 'REFERENCEAXIS'



### **Character Format**

'GOTHIC'  
'UPPERCASE'  
'ITALIC'  
'LOWERCASE'

### **Character Size**

'MEDIUM'  
'SMALL'  
'LARGE'  
'EXTRALARGE'

### **Device Routing**

'PLOTDEVICE'  
'MESSAGEDEVICE'

### **Device Selection**

'ALLDEVICES'  
'PRIMARYDEVICE'  
'ALTERNATEDEVICE'

### **Graphic Input**

'KEYBOARD'	'ECHO'
'CURSORS'	'NOECHO'
'LIGHTPEN'	'ORMODE'
'JOYSTICK'	'ANDMODE'
'BALL'	'PROCEED'
'MOUSE'	'WAIT'

### **Character Type**

'HARDWARE'  
'SOFTWARE'

### **Error Conditions**

'ERROR OUTPUT'	'DUMP'	'NOABORT'	'TRAIL'
'SUPPRESSERRORS'	'ABEND'	'NODUMP'	
'DEFERERRORS'	'ABORT'	'NOTRAIL'	

### **Structure Definition**

'BUILD'  
'NOBUILD'

### **Structure Building**

'EXECUTE'  
'NOEXECUTE'

### **Structure File Manipulation**

'APPEND'                      'REPLACE'

'IGNORE'  
'MULTIPLE'

'REWIND'  
'SINGLE'

#### **Structure Editing**

'COMPRESSED'  
'EXPANDED'

#### **Axis Scaling**

'AUTOSCALE'  
'FULLSCALE'  
'OWNSCALE'

#### **Axis Scale Existence**

'NEWSCALE'  
'OLDSCALE'

#### **Segment Pickability**

'DESENSITIZE'  
'SENSITIZE'

#### **Axis Existence**

'XYAXES'	'NOAXES'
'XAXIS'	'XYZAXES'
'YAXIS'	'XZAXES'
'YZAXES'	'ZAXIS'

#### **Axis Positioning**

'EDGEAXIS'	
'ZEROAXIS'	
'PENAXIS'	
'XEDGEYZEROAXIS'	'YEDGEXZEROAXIS'
'XZEROYEDGEAXIS'	'YZEROXEDGEAXIS'

#### **Numeric Labels**

'BESTFORMAT'  
'FORMAT'  
'GFORMAT'

#### **Label Positioning**

'NEGATIVESIDE'	'PERPENDICULARLABELS'
'PARALLELLABELS'	'POSITIVESIDE'
'XYPLANE'	'XZPLANE'
'YPLANE'	

#### **X Axis Type**

'LINAXIS'  
'LOGAXIS'  
'LNAXIS'

### **Y Axis Type**

'LINYAXIS'  
'LOGYAXIS'  
'LNYAXIS'

### **X Axis Labels**

'XNUMERIC'  
'XALPHANUMERIC'  
'XBOTHLABELS'  
'NOXLABEL'

### **Y Axis Labels**

'YNUMERIC'  
'YALPHANUMERIC'  
'NOYLABEL'  
'YBOTHLABELS'

### **Z Axis Type**

'ZLOGARITHMIC'

### **Z Axis Label**

'ZNUMERIC'  
'ZALPHANUMERIC'  
'NOZLABEL'  
'ZBOTHLABELS'

### **Axis Type**

'TICAXIS'  
'PLAINAXIS'  
'GRIDAXIS'

### **Three Dimensional Windowing**

'SITEPOINT'  
'VIEWPOINT'

### **Windowing**

'CWINDOWING'  
'NOWINDOWING'  
'RWINDOWINE'

### **Text Output**

'ACENTER'  
'NOCENTER'  
'NOSCRIPT'  
'XYZCOORDINATES'  
'XYCOORDINATES'  
'2DCOORDINATES'

'NOSUBSCRIPTING'  
'SUBSCRIPT'  
'SUPERSCRIP'T'  
'INTEGER'  
'TEXT'  
'3DCOORDINATES'

'UPPERCASE'  
'HORIZONTAL'  
'VERTICAL'  
'REAL'



### Menuing Option

'EXTENDEDMENU'  
'STANDARDMENU'

### Origin Inclusion

'NOORIGIN'  
'ORIGIN'  
'PENORIGIN'

### Device Space Clipping

'NOCLIP'  
'ZCLIP'

### High-Level Plotting Option

'NONUNIFORM'  
'NOSTOPDISTORTION'  
'STOPDISTORTION'  
'UNIFORM'

### Coordinate Repeat Option

'NOREPEAT'	'NOZREPEAT'	'YCONSTANT'	'ZREPEAT'
'NOXREPEAT'	'XCONSTANT'	'YREPEAT'	
'NOYREPEAT'	'XREPEAT'	'ZCONSTANT'	

### Transformation Type

'ORTHOGONAL'  
'PERSPECTIVE'

### Axis Orientation

'XNEGATIVE'	'YNEGATIVE'	'ZNEGATIVE'
'XPOSITIVE'	'YPOSITIVE'	'ZPOSITIVE'

### Three Dimension Viewing

'XYVIEW'	'XZVIEW'
'XYZVIEW'	'YZVIEW'

### Rotation Application Order

'XYZ'	'YZX'
'XZY'	'ZXY'
'YXZ'	'XYX'

### Time Axis Scaling

12Hour  
13Week  
24Hour

## APPENDIX F

### UQUERY OPTIONS

Type	Query Name	Returns (Default in Parentheses)
USET	'ABUTTING'	Page abutting mode ('NONABUTTING')
USET	'ACENTERING'	Alphanumeric centering ('NOCENTERING')
USET	'ADJUSTMENT'	3D Axis view adjustment option (plane axis plotted on view port or viewed from current view point) ('XYZVIEW')
USET	'ANGULARUNITS'	Current user angular units ('DEGREES')
USET	'ANGLE OF TEXT'	Current angle of text output (0.)
UPSET	'ASPECTRATIO'	3D Display Surface aspect ratio
UPSET	'ATTENTION QUEUE SIZE'	Attention queue size (100.)
UPSET	'BACKGROUND COLOR'	Background color index (Black = 0.)
USET	'BLENDMODE'	Color Blending mode ('SUBTRACTIVE')
USET	'BLINKRATE'	Blink rate ('NOBLINK')
UPSET	'BRIGHTNESS'	Display intensity (60%)
USET	'BUILD'	3D Structure build mode flag ('NOBUILD')
UPSET	'CHARACTER'	Current system character as Hollerith string (*)
USET	'CLIPPING'	Device space clipping flag ('NOCLIP', 'CLIP' or 'INVERTED')
USET	'COLOR'	Current color index (device dependent)
UPSET	'COPY DELAY'	Copy delay time (device dependent)
UPSET	'CSPACING'	Character spacing mode ('HORIZONTAL')
USET	'CURVE'	Current curve approximation mode ('CONTINUOUS')
UPSET	'DASH'	Numeric value corresponding to current dashline specification (56.)
USET	'DESCRIPTION'	Current axis option ('TICAXES')
USET	'DETECTABILITY'	Detectability of new segments ('DESENSITIZED')
USET	'DIMENSION'	3D 2D or 3D coordinate terminator switch ('2DCOORDINATES')
UPSET	'DISTANCE'	3D Distance from viewport to screen plane (150.)
USET	'EDIT'	3D Data structure edit mode ('COMPRESS')
USET	'ERRORMODE'	Error presentation (ERROR)
USET	'EXECUTE'	3D Structure building visibility ('EXECUTE')
USET	'EXISTENCE'	Current axis existence option ('XYZAXES')
UPSET	'FACTOR'	Scale factor for GCS created software symbols (1.)
USET	'FITMODE'	Curve fitting mode for autoplotting ('NOFITTING')

USET	'FNAME MODE'		Frame naming mode ('FNAME')
UPSET	'FNTFILE NUMBER'		Font file number (0.)
UPSET	'FONT NAME'		Font name ('GCS')
USET	'FORMAT'		Text number format for numeric labelling ('BESTFORMAT')
USET	'GAPMODE'		Gapped line mode ('UINTERRUPTED')
UPSET	'GREYSCALE'		Numeric value indicating current grey level (device dependent)
UPSET	'GRID'	3D	Numeric value indicating grid axis type option (0.)
USET	'HANDEDNESS'	3D	Left or right handed coordinate system ('RIGHTHANDED')
UPSET	'HARDWARE'		Current hardware character size ('SMALL', device dependent)
UPSET	'HORIZONTAL'		Current horizontal software character position size (5.)
UPSET	'INFILE'		Graphics input file designation (computer system dependent)
USET	'INPUT'		Graphics input device medium
USET	'ITALICIZATION'		Italicization mode ('NOITALICS')
UPSET	'LABELANGLE'		Label angle around perpendicular (0.)
USET	'LETTERTYPE'		Character type ('HARDWARE')
UPSET	'LEVEL'	3D	Subscript/superscript spacing level (1.)
UPSET	'LIBRARY'	3D	Data structure library file mode (0.)
UPSET	'LIMIT'	3D	Number of errors before automatic stop. 0 means no limit (0.)
USET	'LINEOPTION'		Current line option setting ('LNULL')
UPSET	'LOWERCASE'		Lowercase shift character as Hollerith string ('>')
USET	'LOGARITHMIC'	3D	Axes selected for logarithmic transform (logarithmic transform switch) ('NOLOGSCALING')
USET	'LOGTIME OF APPLICATION'		Time of application of logarithmic scaling ('LOGSYSTEM COORDINATES')
USET	'LOGTYPE'	3D	Flag indicating when logarithmic scaling performed. ('LOGSYSTEM', 'LOGUSER', 'LOGOBJECT')
USET	'MAPPINGTYPE'	3D	3D to 2D mapping type ('PERSPECTIVE')
USET	'MENUTYPE'	3D	Menu board type ('STANDARD')
USET	'MERGE'	3D	Structure merge mode switch ('IGNORE')
USET	'MESSAGEDEVICE'		Destination of alphanumeric I/O ('PLOTDEVICE')
USET	'MODE'		Current coordinate mode ('ABSOLUTE')
USET	'NUMERIC'	3D	Numeric labels parallel or perpendicular to axis ('PARALLEL')
USET	'ORDER'	3D	3D coordinate system rotation application order ('ZYX')



UPSET	'ORIENTATION'		Angular orientation of display of GCS created symbols (0.)
USET	'ORIGIN'		Forced origin switch for axis scaling ('ORIGIN')
UPSET	'OUTFILE'		File number of graphics output file (computer system dependent)
USET	'OUTPUTDEVICE'		Graphical output destination device ('ALLDEVICES')
USET	'PLANE'	3D	Axis label plane ('XYPLANE')
USET	'PLOTSCALE'		Scale option ('NEWSCALE')
UPSET	'POLYNOMIALDEGREE'		Current degree of polynomial fit for curve fitting (5.)
USET	'POSITION'	3D	Current axes positioning ('EDGEAXES')
UPSET	'PRECISION'		Number of digits of precision to be displayed for real numbers. (4.)
USET	'PROJECTION TYPE'		Type of projection ('PERSEPCTIVE')
UPSET	'READ FILE'		File designator for non-graphic input (computer system dependent)
USET	'REPEAT'	3D	Data structure invocation option ('SINGLE' or 'REPEAT')
USET	'REWIND'	3D	Structure file rewind mode ('REWIND')
USET	'SCALE'		Axis scale option ('AUTOSCALE')
USET	'SCRIPT'	3D	Subscript/superscript control ('NOSCRIP')
USET	'SECURITY LEVEL'		Control of security banners ('UNSECURED')
USET	'SENSITIVITY'	3D	Light pen sensitivity switch ('DESENSITIZE')
USET	'SIDE'	3D	Side of axes on which labels will appear ('NEGATIVE')
USET	'SIZE'		Hardware character size ('SMALL')
UPSET	'SLANT'	3D	Software character italic slant angle (18. degrees)
USET	'SPACE'		Coordinate space ('VIRTUAL')
UPSET	'SPAN'	3D	Angular span for pie charts (360. degrees)
UPSET	'START'	3D	Pie chart starting angle (0.)
USET	'STOP'		Error stopping control ('NOABORT')
USET	'STORAGEMODE'		Segment/Frame retention mode ('RETAINED')
UPSET	'STRUCTURE LIMIT'		Maximum number of structures which can be defined (100.)
UPSET	'SUBSCRIPTCHARACTER'	3D	Current subscript shift character
UPSET	'SUPERSCRIPCHARACTER'	3D	Current superscript shift character
UPSET	'SYMBOL INDEX'		Choice of marker (0.)
USET	'SYSTEM'		Current coordinate system ('SYSTEM')
UPSET	'TABHORIZONTAL'		Location of current horizontal tab stop (10.)
UPSET	'TABVERTICAL'		Location of current vertical tab stop (10.)
UPSET	'TERMINATOR'		GCS string termination character

USET	'TEXT'		Textual I/O mode ('TEXT')
UPSET	'TICINTERVAL'		Length of UPEN ticintervals (10.)
UPSET	'TICLENGTH'		Length of UPEN ticintervals (10.)
UPSET	'TICMINUS'		Clockwise tic mark size (0.05)
UPSET	'TICPLUS'		Counter-clockwise tick mark size (0.05)
UPSET	'TICX'		X axis tic interval (0.)
UPSET	'TICY'		Y axis tic interval (0.)
UPSET	'TICZ'	3D	Z axis tic interval (0.)
USET	'TIME'		Time series plotting period ('DATES')
USET	'TYPE'		Type of coordinates ('RECTANGULAR')
USET	'UNIFORM'	3D	High level plotting option ('NONUNIFORM')
USET	'UNITS'		Device space units ('INCHES')
USET	'UPAXIS'	3D	Coordinate system viewport vertical axis ('ZPOSITIVE')
UPSET	'UPPERCASE'		Uppercase shift character as Hollirith string ('<')
USET	'USER'		User coordinate system switch
UPSET	'VERTICAL'	3D	Vertical software character position switch
USET	'VIEWPORT'		Viewport distance base switch ('SITE')
USET	'VISIBILITY'		Framed/segment creation visibility mode ('VISIBLE')
UPSET	'WIDTH OF LINES'	3D	Width of lines (0.)
USET	'WINDOW'		Window type ('NOWINDOW')
UPSET	'WRITE FILE'		Non-graphic alphanumeric output file (Computer system dependent)
UPSET	'XBASE'		Base of log scaling along x-component (real number or string 'E') (10.)
USET	'XLABEL'		X axis labelling option ('XNUMERICLABEL')
USET	'XLOGARITHMIC'		X axis linearity option
UPSET	'XPERCENTAGE OF CHARACTER SPACE'		Portion of horizontal character space occupied by character (0.65)
USET	'XREPETITION MODE'		X component repetition mode for plotting ('NOXREPEAT')
UPSET	'XROTATION'		Rotation factor around X axis for structure invocation (0.)
UPSET	'XSCALING'		Scaling factor along X axis for structure invocation (1.)
UPSET	'XSPECIFICATION UNIT'		Number of XSPECIFICATION units in device space (1000.)
USET	'XSIZE'		Horizontal hardware character position size (device dependent)
UPSET	'XTITLE'		X axis alphanumeric label ('X')

UPSET	'YBASE'	3D	Base of log scaling along Y-component (real number or string 'E') (10.)
USET	'YLABEL'		Y axis labelling option ('YNUMERICLABELS')
USET	'YLOGARITHMIC'		Y axis linearity option
UPSET	'YPERCENTAGE OF CHARACTER SPACE'		Portion of vertical character space occupied by character (0.65)
USET	'YREPETITION'		Y component repetition made for plotting ('NOYREPEAT')
UPSET	'YROTATION'		Rotation factor around Y axis for structure invocation (0.)
UPSET	'YSCALING'		Scaling factor along Y axis for structure invocation (1.)
UPSET	'YSIZE'		Vertical size for hardware characters (device dependent)
UPSET	'YSPECIFICATIONUNITS'		Number of Y specification units in device space (1000.)
UPSET	'YTITLE'		Y axis alphanumeric label
UPSET	'ZBASE'	3D	Base of log scaling along Z axis (10.)
USET	'ZCLIP'		Hither/yon clipping mode ('NOZCLIPPING')
USET	'ZLABEL'	3D	Z axis label option ('ZNUMERICLABELS')
USET	'ZREPETITION MODE'		Z component repetition mode for plotting ('NOZREPEAT')
UPSET	'ZROTATION'		Rotation factor around Z axis for structure invocation (0.)
UPSET	'ZSCALING'		Scaling factor along Z axis for structure invocation (1.)
UPSET	'ZSPECIFICATIONUNITS'		Number of Z specification units in device space (1000.)
UPSET	'ZTITLE'	3D	Z axis title
UPSET	'ZVALUE'	3D	Numeric default Z-value for 2D coordinates (0.)



## APPENDIX G

### GCS ERROR CODES

The following is a list of the currently defined error codes in GCS:

- 01 — Invalid key word specification to USET.
- 02 — Invalid key word specification to UPSET.
- 03 — No plot files found.
- 04 — Invalid option for UPRNT1.
- 05 — Invalid UDOIT system.
- 06 — Invalid view port boundaries.
- 09 — Invalid UINPUT option.
- 10 — UMARGN argument list out of order/Boundary specification invalid.
- 11 — UMARGN boundary outside of physical device boundary.
- 12 — UWINDO argument list out of order/Boundary specifications invalid.
- 13 — UDAREA argument list out of order/Boundary specifications invalid.
- 14 — UDAREA boundary outside of physical device boundary.
- 15 — UCLIP invalid argument list.
- 16 — UCLIP boundary overlap error.
- 17 — UDIMEN maximum boundary specification invalid.
- 18 — UAXIS argument list XMIN .GT.XMAX and/or YMIN .GT. YMAX.
- 19 — UDAREA provided to UAXIS too small for requested options.
- 20 — UPSET — invalid tic interval. Value .LE.zero specified.
- 21 — UPSET — invalid scale factor. Value must not be equal to zero.
- 22 — UPSET — invalid software character size. Value must not be equal to zero.
- 23 — UPSET — invalid digits of precision. Precision must be greater than zero.
- 24 — UPSET — attempt to set zero or negative scripting — level.
- 25 — UPSET — Invalid light pen correlation value. Value must be.GT.zero.
- 26 — UPSET — Invalid transmission speed. Value must be .GT.zero.
- 27 — UPSET — Invalid Library file code. File codes must be in range 1 to 99.
- 28 — UPSET — Invalid error limit. Value must be greater than zero.
- 29 — UPSET — Invalid error limit. Value must be greater than zero.
- 30 — UFRAME/UFREND frame table full.
- 31 — Failure to call UFREND for previous occurrence of same frame.
- 32 — Failure to call UFREND for another named frame.
- 33 — UFREND called without any frame active.
- 34 — UFREND call not for currently active frame.
- 35 — USHOW/UNSHOW called for undefined frame.
- 36 — USHOW/UNSHOW called while in frame build status.
- 51 — Insufficient data points for desired fit.
- 52 — Input data points exceed fit capacity.
- 53 — Input data out of order or non functional relationship.
- 54 — Insufficient array space to return fitted results.
- 55 — Invalid trend adjustment factor.
- 56 — Invalid polynomial degree for polynomial fit.
- 57 — Invalid number of previous periods for moving average fit.
- 60 — Attempt to build a structure in frame mode.
- 61 — Nested structure call stack overflow.
- 64 — Attempt to redefine existing structure.
- 65 — Structure table overflow.
- 66 — Attempt to activate a structure while another is still active.
- 67 — Structure termination without active structure.
- 68 — Structure termination not for current structure.
- 69 — Attempt to execute an undefined structure.
- 70 — Recursive structure build call.
- 71 — Invalid number of items for UREAD/UINPUT.
- 72 — Attempted input operation from batch device.

- 80 - Attempt to create a secondary axis scale of zero.
- 81 - UAXIS - pen position outside of UDAREA in 'PENAXES'.
- 82 - Attempt to plot log with window bound .LE. 0.0
- 83 - Attempt to move to apply log scaling to coordinate whose value is zero.
- 84 - Log plotting in device space not allowed.
- 85 - UAXIS - AXIS choice requires 0 to 1 within range of X and/or Y AXIS.
- 86 - ULINE/U3LINE - Invalid number of points.
- 91 - UAPEND found zero length string. Terminator placed in first character position.
- 92 - UWAIT - Negative time period specified.
- 93 - UQUERY - Invalid option specification.
- 101 - UHISTO - Insufficient points.
- 102 - UHISTO - Invalid number of bars.
- 103 - UHISTO - Unreasonable window for OWNSCALE.
- 104 - UHISTO - Insufficient UDAREA for options specified.
- 110 - Invalid number of points for UPIE.
- 111 - Invalid data value for UPIE.
- 112 - Invalid max label size for UPIE.
- 113 - Insufficient room for UPIE display.
- 114 - UPIE - ABS(Starting Angle) > -ABS(Ending Angle).
- 115 - UPIE - Too many labels outside of pie.
- 120 - USCATR - Insufficient points specified.
- 121 - USCATR - Invalid limits for logarithmic scatter diagram.
- 122 - USCATR - UDAREA too small for specified options.
- 130 - UBAR - Invalid number of points.
- 131 - UBAR - Invalid label size.
- 132 - UBAR - Invalid data value.
- 133 - UBAR - Insufficient UDAREA for options specified.
- 140 - UCHART - Invalid number of points.
- 141 - UCHART - Invalid label size.
- 142 - UCHART - Invalid number of bars.
- 143 - UCHART - UDAREA too small for specified options.
- 150 - UTAXIS - UDAREA too small for specified options.
- 160 - UTILTY - Invalid action.
- 161 - UTILTY - Save structure not found.
- 162 - UTILTY - Utility operation on structure work file.
- 190 - UVIEW - Viewpoint specified same as view site.
- 191 - UVWPRT - Aperature specified negative or zero in some dimension.
- 192 - UVWPRT - Viewport behind viewer.
- 193 - Attempt to draw through viewpoint.
- 194 - Hither plane behind or at viewpoint.





# WATERWAYS EXPERIMENT STATION

Vicksburg, Miss.

## GRAPHICS COMPATIBILITY SYSTEM (GCS) PROGRAMMER'S REFERENCE MANUAL

for the WES Automatic Data  
Processing Center

1978



## ELECTRONIC COMPUTER PROGRAM ABSTRACT

## TITLE OF PROGRAM

Graphics Compatibility System (GCS)

## PROGRAM NO.

803-F3-R0200

PREPARING AGENCY U. S. Army Engineer Waterways Experiment Station, Automatic  
Data Processing Center, P. O. Box 631, Vicksburg, Miss. 39180

## AUTHOR(S)

West Point Military Academy  
Waterways Experiment Station (ADPC)

## DATE PROGRAM COMPLETED

1975

Revised 1978

## STATUS OF PROGRAM

PHASE

STAGE

OP

## A. PURPOSE OF PROGRAM

GCS is a general-purpose, device-independent computer graphics package keyed to the user.

## B. PROGRAM SPECIFICATIONS

GCS is a collection of ANSI standard FORTRAN subroutines invocable by a user's FORTRAN program.

## C. METHODS

GCS's capabilities range over a powerful set of functions. It can produce a simple line-drawing (with over 40 types of lines), or handle comprehensive general-purpose axis-creation and automatic clipping; ARC and Conic generation; graphics input; drafting multilevel, secondary-coordinate system definition and characters; and italicized software characters. All plotting and other positional output can be done in the user's own units without his having to know anything about the specific nature of his terminal.

## D. EQUIPMENT DETAILS

GCS is operational on the Honeywell 635 and supports the Tektronix 4010/4014, Computer 400/15, IMLAC PDS-1D, TSP Analog pen plotter, Datapoint 3300, and alphanumeric printing terminals. Nationwide field testing has included, or will include, such equipment as the Univac 1108, IBM S/360-370, DEC PDP 10, HP 3000, and CDC 3500 and 6000 series. A tape must be provided by interested parties if a copy of the GCS system is desired.

## E. INPUT-OUTPUT

Input-Output is accomplished by invoking GCS subroutines.

## F. ADDITIONAL REMARKS

Manuals available:

1. Primer on Computer Graphics Programming.
2. GCS Programmer's Reference Manual.

## **CHAPTER I**

### **INTRODUCTION**

Contained within this section are the detailed descriptions and usage instructions of all of the user callable routines which make up the GCS graphics system, and a discussion of the various options available to the user.

GCS is available in two versions, a two-dimensional version, and a three-dimensional version.

The three dimensional version is a sophisticated, highly capable package having such features as graphical data structures, text processing, and of course, three-dimension graphics. The two-dimensional version is the older, less sophisticated version. While possessing fewer capabilities, it is still supported, since it offers the benefits of a smaller central memory requirement, and a higher execution speed, while still fulfilling most of the normal graphics requirements.

In general, the two-dimension version is a proper subset of the three dimension version meaning that all elements of two-dimension GCS are contained within the three dimensional version.

Those routines or options which are a part of the three-dimension version, and not a part of the two-dimension version are indicated by the notation '3D only.'

## CHAPTER II

### THE GRAPHICS STATUS AREA (GSA) AND USET/UPSET/UQUERY OPTIONS

#### The Graphics Status Area

Central to the structure of GCS is the concept of a distributed executive capability provided by the Graphics Status Area (GSA). This is a labeled COMMON area in which is stored information concerning the state of the graphics system, information about the characteristics of both the terminal and the computer system, and information about the current settings of GCS user-controlled variables. The GSA serves as a means of communication between the various subroutines which comprise the Graphics Compatibility System. Much of the power of GCS arises from the ability of every GCS routine to access any of the information about the status of the system conveniently and efficiently. One of the immediate results of having a status area is the great reduction in the number of arguments in the calling sequences of the routines, thus improving the efficiency of the system. It also makes it much easier for the user to remember the calling sequence of the different routines when writing application software.

The variables of the GSA can be set in any of three ways: they may be established during the execution of USTART with values taken from a predefined table; they may be set by the user through USET, UPSET, or other setting routines; or they may be set by GCS routines during execution. USTART will load the variables with their initial default setting. The user sets these variables under his control in USET and UPSET and through subroutine arguments for such routines as UWINDO, UMARGN, UDAREA, UCOSYS, and UROTAT. The user may also restore the GSA variables to the previous state through use of any of the 'UNSAVE' routines. GCS itself maintains many variables which keep track of such items as beam/pen position, and various status change flags. The parameters contained within the GSA can be obtained by the user by a call to UQUERY.

The ability to change GSA variables selectively in accordance with related subroutine options is one of the many attractive and powerful features of GCS. This chapter contains two tables which present descriptive grouping of USET and UPSET options. Table 1 gives a list of those USET options which are most frequently used. This list may be used as a quick reference guide to the most popular USET options. Table 2 gives the default conditions that are automatically preset as each call to GCS is initiated. This concept of default values is immensely valuable and convenient to both the beginning programmer and the experienced programmer who doesn't want to worry about the setting of the GSA variables. The complete list of options can be formed in the writeups associated with the USET/UPSET/UQUERY subroutines.

#### USET Options

TABLE 1

##### USET OPTIONS MOST FREQUENTLY USED:

TO REQUEST ABSOLUTE COORDINATE PLOTTING (DEFAULT):

USET ('ABSOLUTE')	USET ('ABSO')
USET ('ABSb')	

TO REQUEST RELATIVE OR INCREMENTAL COORDINATE PLOTTING:

USET ('RELATIVE')	USET ('RELA')
-------------------	---------------

NOTE: b - is a blank or space



TO REQUEST RECTANGULAR COORDINATES (DEFAULT):

    USET ('RECTANGULAR')      USET ('RECT')

TO REQUEST POLAR COORDINATES:

    USET ('POLAR')              USET ('POLA')

TO REQUEST ANGULAR VALUES IN DEGREES (DEFAULT):

    USET ('DEGREES')          USET ('DEGR')

TO REQUEST ANGULAR VALUES IN RADIANS:

    USET ('RADIANS')          USET ('RADI')

TO REQUEST PLOTTING IN VIRTUAL SPACE (DEFAULT):

    USET ('VIRTUAL')          USET ('VIRT')

TO REQUEST PLOTTING IN DEVICE SPACE:

    USET ('DEVICE')            USET ('DEVI')

TO REQUEST DEVICE SPACE PLOTTING UNITS IN INCHES (DEFAULT):

    USET ('INCHES')            USET ('INCH')

TO REQUEST DEVICE SPACE PLOTTING IN RASTER UNITS:

    USET ('RASTERUNITS')      USET ('RAST')

TO REQUEST DEVICE SPACE PLOTTING IN FONT (CHARACTER SPACE) UNITS:

    USET ('FONTUNITS')        USET ('FONT')

TO REQUEST DEVICE SPACE PLOTTING IN PERCENT UNITS:

    USET ('PERCENTUNITS')    USET ('PERC')

TO REQUEST PLOTTING OF A VISIBLE LINE (DEFAULT):

    USET ('LINE')

TO REQUEST PLOTTING OF INVISIBLE LINES:

USET ('MOVE')	
USET ('NOLINE')	USET ('NOLI')
USET ('NOBLINE')	USET ('NOBL')

NOTE: b - is a blank or space

TO REQUEST PLOTTING OF LINES WITH ARROW TERMINATORS:

    USET ('ARROW')            USET ('ARRO')

TO REQUEST PLOTTING OF DASHED LINES:

    USET ('DASH')

TO REQUEST PLOTTING OF TIC LINES:

    USET ('TICLINE')          USET ('TICL')

TO REQUEST PLOTTING OF LINES WITH ONLY ENDPOINTS VISIBLE:

    USET ('POINT')            USET ('POIN')

TO REQUEST PLOTTING OF LINES COMPOSED OF CHARACTERS:

    USET ('ALPHANUMERICLINES')  USET ('ALPH')

TO REQUEST PLOTTING OF INVISIBLE LINES WITH CHARACTERS AT THEIR  
ENDPOINTS:

    USET ('CHARACTER')         USET ('CHAR')

TO REQUEST CHARACTER OUTPUT IN THE FORM OF HARDWARE CHARACTERS  
(DEFAULT):

    USET ('HARDWARE')          USET ('HARD')

TO REQUEST CHARACTER OUTPUT IN THE FORM OF SOFTWARE CHARACTERS:

    USET ('SOFTWARE')          USET ('SOFT')

TO REQUEST UPRINT/UWRITE OUTPUT IN TEXT FORMAT (DEFAULT):

    USET ('TEXT')

TO REQUEST UPRINT/UWRITE OUTPUT IN REAL FORMAT:

    USET ('REALNUMBER')        USET ('REAL')

TO REQUEST UPRINT/UWRITE OUTPUT IN INTEGER FORMAT:

    USET ('INTEGER')          USET ('INTE')

TO REQUEST PLOTTING WITH RESPECT TO THE SYSTEM AXIS (DEFAULT):

    USET ('SYSTEM')            USET ('SYST')

---

NOTE: b - is a blank or space

TO REQUEST PLOTTING WITH RESPECT TO THE USER AXIS:

USET ('USERAXIS')

USET ('USER')

TO REQUEST A LINE TYPE/CHARACTER TERMINATOR COMBINATION:

USET ('X\$bb')

USET ('X\$')

Where

\$ IS THE DESIRED CHARACTER TERMINATOR

AND

X, THE LINE TYPE SPECIFICATION, IS AS FOLLOWS:

A ALPHANUMERIC LINE  
D DASHED LINE  
L SOLID LINE  
N NULL OR INVISIBLE LINE  
T TIC LINE  
b denotes blank

TABLE 2

GCS DEFAULT CONDITIONS

This table discusses those options which are present in the Graphics Status Area as default options. After a call to Subroutine USTART, the Graphics Compatibility System is set to the default conditions as indicated below. The default options can be divided into two groups: Basic Plotting Options and High Level Plotting Options.

I. Default Basic Plotting Options

Plotting is done in 'RECTANGULAR' and 'ABSOLUTE' coordinates on the 'SYSTEM' coordinate axis. The 'USER' coordinate axes are identical to the system axis. Plotting is done in 'VIRTUAL' space with a virtual window whose limits are from 0.0 to 100.0 in the X direction, and from 0.0 to 100.0 in the Y direction. The virtual window is mapped into a display area which is the largest square area on the device display surface. The right hand edge of the square corresponds to the top edge of the display surface.

Character output in GCS will be, by default, in 'HARDWARE' character format, of type 'GOTHIC' and of 'MEDIUM' size. If 'SOFTWARE' characters are requested via USET then the default horizontal size is five (5.0) virtual units, and the default vertical size is seven (7.0) virtual units.

By default, angular units for angular specifications are in 'DEGREES'. If the user switches to 'DEVICE' space, then all length or distance units (i.e. from UPRINT or UREAD), the data will be assumed to be in 'TEXT' mode. The alphanumeric output defaults to the 'PLOT' device if possible, and any graphic output will go to all devices in a cluster. Graphic input is from the primary input device; the number of digits of precision for numeric output is four (4); and GCS detected errors are signalled as they occur.

A line which is drawn by a subroutine in GCS is considered to consist of two parts; a line type and a line terminator. The line type may be solid (visible), ticked, null (invisible), dashed, or alphanumeric. The line may be terminated by an arrow, a back arrow, double arrows, a character, a point, a symbol, a set of coordinate values, or nothing at all. The default line type in GCS is 'SOLID' with 'NULL' terminators ('LNULL'). If 'TICLINES' are requested via USET option, then the default tic interval is ten (10.0) virtual units. It is the user's responsibility to insure that the tic interval is appropriate if he switches to

NOTE: b - is a blank or space



'DEVICE' space or alters the default virtual window setting or the default display area setting. If 'DASHLINES' are requested, then the default dash specification (56.) will result in a dashed line which is alternately light and dark, in increments of approximately 0.075 inches. If a 'CHARACTER' line terminator is requested but no character is specified by way of Subroutine UPSET, then the character asterisk (\*) is used. Similarly, the asterisk is used to compose the line type if 'ALPHANUMERIC' lines are requested.

## II. Default High Level Plotting Options

For each call to UPLLOT or UPLLOT1, the data values which represent the curves are examined, and a 'NEWSCALE' is created. UAXIS will be invoked to create 'XYAXES'. The data values will be examined, and appropriate limits for the data established. Numeric labels only will be output for the X axis and the Y axis. The values which appear at the tic marks on the axes will be 'neat' numbers, and the axes will be positioned at the 'EDGE' of the plot. Both the X axis and the Y axis will be drawn in a linear coordinate space. The axis lines will be ticced. If a time series axis is plotted by invoking Subroutine UTAXIS, the default interval for the X axis will be 'DAILY'. No curves will be fit to the data values, but if 'FITPOLYNOMIAL' is requested, then subroutine UPLLOT will attempt to fit a fifth degree polynomial to the data.

## III. Summary

COORDINATE SPACE:	'VIRTUAL'
COORDINATE TYPE:	'ABSOLUTE'
COORDINATE SYSTEM:	'RECTANGULAR'
COORDINATE AXIS:	'SYSTEM'
VIRTUAL WINDOW:	0.0 TO 100.0 X DIRECTION 0.0 TO 100.0 Y DIRECTION
DISPLAY AREA:	Largest square area which is right-justified on the display surface of the device.
DEVICE SPACE UNITS:	'INCHES'
ANGULAR UNITS:	'DEGREES'
LINE TYPE:	'LINE' or 'LNULL'
SYSTEM CHARACTER:	asterisk (*)
TIC INTERVAL:	10.0 virtual units
DASH SPECIFICATION:	56.
CHARACTER TYPE:	'HARDWARE'
CHARACTER FONT:	'GOTHIC'
CHARACTER SIZE:	'MEDIUM'
SOFTWARE CHARACTER SIZE:	5.0 virtual units horizontal 7.0 virtual units vertical
OUTPUT FORMAT:	'TEXT'
ALPHANUMERIC MARGINS:	Device display surface boundaries.
DIGITS OF PRECISION:	4
ERROR HANDLING:	'IMMEDIATEOUTPUT'
AXIS SCALING:	'AUTOSCALE'
AXIS LABELING:	'XNUMERICLABEL' 'YNUMERICLABEL'
AXIS POSITIONING:	'EDGEAXIS'
AXIS TYPE:	'TICAXIS'
AXIS EXISTENCE:	'XYAXES'
AXIS COORDINATES:	'LINXAXIS' 'LINYAXIS'
TIME SERIES AXIS SCALE:	'DAILY'

## CHAPTER III

### GCS USER SUBROUTINES

This chapter provides the detailed description of each user callable subroutine. Each routine is presented in the following manner: The FUNCTION paragraph gives a brief statement of the main purpose of the subroutine. The CALLING SEQUENCE is presented with an explanation of each parameter required for the subroutine. Any USET or UPSET options that will affect the operation of the subroutine are listed under the OPTIONS paragraph. The COMMENTS paragraph provide the detailed description of all of those features considered to be both unique and important to each specific subroutine. On the following pages is listing of all of the user subroutines. The features that are only calculated in the three dimensional version are specified by italics.

## ALPHABETICAL LISTING OF GCS SUBROUTINES

UAIN	Accepts one character from the terminal CALL UAIN (ICHAR)
UALPHA	Insures that terminal is in alphanumeric mode CALL UALPHA
UAOUT	Outputs a character at current pen position subject to margining CALL UAOUT (ICHAR)
UAPEND	Adds GCS string terminator to character string CALL UAPEND (COUNT,DATAIN,DATOUT)
UARC	Draws an arc from current pen position CALL UARC (X,Y,ANGLE)
UASPCT	Forces the display dimensions to satisfy the specified aspect ratio CALL UASPCT (RATIO)
UAVERG	Fits a moving average curve to time series data CALL UAVERG (ARRAY,POINTS,FCST,PERIOD)
UAXIS	Draws axes with appropriate numeric and alphanumeric labeling CALL UAXIS (XMIN,XMAX,YMIN,YMAX)
UBAR	Draws a bar chart with appropriate numeric and alphameric labels CALL UBAR (ARRAY,PTS,LABELS,SIZE)
UBELL	Sounds the audible alarm at the terminal CALL UBELL
UCALL	Invokes a graphic data structure in two dimensions CALL UCALL (NAME,DX,DY,SX,SY,ANGLE)
UCHAR	Draws a grouped bar chart for multi-valued data CALL UCHAR (ARRAY,GROUPS,BARS,LABELS,YMAXL)
UCLOSE	Closes the current open frame/segment CALL CLOSE (SEGNAM)
UCONIC	Draws generalized conic sections CALL UCONIC (X,Y,P,E,THETA1,THETA2)



<b>UCONTR</b>	Draws contours on regular array of data CALL UCONTR (Z,X,Y,A,FX,FY,CURVE,FN)
<b>UCOSYS</b>	Creates a user coordinate plotting system CALL UCOSYS (DX,DY,SX,SY,ANGLE)
<b>UCOUNT</b>	Counts number of characters in character string CALL UCOUNT (DATA,COUNT)
<b>UCRCLE</b>	Draws a circle whose center location and radius are specified CALL UCRCLE (X,Y,RADIUS)
<b>UDAREA</b>	Sets the device display area associated with user window CALL UDAREA (XMIN,XMAX,YMIN,YMAX)
<b>UDELET</b>	Deletes a currently-defined frame/segment CALL UDELETE (SEGNAM)
<b>UDIMEN</b>	Adjusts physical boundaries of output device (alters aspect rates) CALL UDIMEN (XMAX,YMAX)
<b>UDOIT</b>	Perform various page layout functions CALL UDOIT (ACTION)
<b>UDRAW</b>	Draws solid line vector CALL UDRAW (X,Y)
<b>UDRIN</b>	Performs the input requested graphic operation and returns request CALL UDRIN (X,Y,ICHAR)
<b>UEND</b>	Terminates graphic operations and positions pen in home position CALL UEND
<b>UERASE</b>	Erases the screen or requests a clean plotting surface CALL UERASE
<b>UERROR</b>	Returns listing of source records with GCS error commentary CALL UERROR (ERLAST,TOTAL)
<b>UFLUSH</b>	Insures that visual display reflects all net program graphical output CALL UFLUSH

UFRAME	Defines the start of a named set of graphical commands CALL UFRAME (NAME)
UFREND	Defines the end of a named set of graphical commands CALL UFREND (NAME)
UGRIN	Gets coordinates and a character from terminal and returns them CALL UGRIN (X,Y,ICHAR)
UHISTO	Draws a histogram with appropriate numeric and alphanumeric labels CALL UHISTO (ARRAY,PTS,BARS)
UHOME	Moves beam to home position CALL UHOME
UIMAGE	Applies general 2-D image transformations to 'retained' segments CALL UIMAGE (X,Y,SX,SY,R,SEGNAM)
UINPUT	Inputs alphanumeric information from the current position CALL INPUT (DATA,COUNT,FLAG,OPTION)
UINVOK	Invokes a GCS structure at the current position CALL UINVOK (NAME)
ULINE	Draws a curve connecting two arrays of points in two spaces CALL ULINE (X,Y,PTS)
ULINFT	Calculates least squares linear fit to points provided CALL ULINFT (X,Y,XN,S,YI)
ULOOK	Establish portion display area onto which corresponding portion of current virtual space viewport will be mapped CALL ULOOK (XMIN,XMAX,YMIN,YMAX)
ULSTSQ	Calculates least squares polynomial fit to points provided CALL ULSTSQ (X,Y,XN,COEFF)
UMARGN	Sets the left and right, top and bottom alphanumeric window boundaries CALL UMARGN (XLEFT,XRIGHT,YBOTTM,YTOP)
UMENU	Menu board generating routine CALL UMENU (POINTS,LABELS,CHOICE)

<b>UMODFY</b>	Modifies setting of segment attributes CALL UMODFY (SEGNAM,NAMAT,ATVALU)
<b>UMOVE</b>	Moves the pen to position specified by input arguments CALL UMOVE (X,Y)
<b>UNSAVE</b>	Restores all variables of the graphic status area CALL UNSAVE (ARRAY)
<b>UNSHOW</b>	Causes the named frame of graphical information to be made invisible CALL UNSHOW (NAME)
<b>UNSVPN</b>	Restores all pen related variables in the graphics status area CALL UNSVPN (ARRAY)
<b>UNSVTR</b>	Restores coordinate system related variables in the graphics status area CALL UNSVTR (ARRAY)
<b>UOPEN</b>	Opens a segment CALL UOPEN (SEGNAM,SEGTP)
<b>UORIGN</b>	Creates a user coordinate system at the current beam/pen position CALL UORIGN
<b>UOUTLN</b>	Draws a box around the user's display area CALL UOUTLN
<b>UPAUSE</b>	Suspends execution until one character is entered from keyboard CALL UPAUSE
<b>UPEN</b>	Draws a line from current pen position to given coordinates CALL UPEN (X,Y)
<b>UPEN1</b>	Sets one 'USET' option for this call only before executing pen movement CALL UPEN1 (X,Y,OPTION)
<b>UPIE</b>	Draws a pie chart with appropriate numeric and alphameric labels CALL UPIE (ARRAY,PTS,LABELS,SIZE)
<b>UPLACE</b>	Applies 2-D translation image transformation to 'retained' segments CALL UPLACE (X,Y,SEGNAM)



UPLOT	General purpose multi-curve plotting routine CALL UPLOT (X,Y,CURVES,PARRAY,OPTION)
UPLOT1	Plots a single curve CALL UPLOT1 (X,Y,PTS)
UPLYGN	Draws a regular polygon CALL UPLYGN (X,Y,PTSIN,RADIUS)
UPOINT	Defines point which, together with two end points of a given line, defines the plane for the terminator and tic line CALL UPOINT (X,Y,Z)
UPOST	Insures that only defined, visible segments are displayed CALL UPOST
UPRINT	Prints information in hardware or software characters CALL UPRINT (X,Y,INPUT)
UPRNT1	Allows alphanumeric output at current position with specified option CALL UPRNT1 (DATA,OPTION)
UPSET	Changes setting in the gas which requires a parameter value to be set CALL UPSET (OPTION,VALUE)
UQUERY	Obtains current value of specified variable in GSA CALL UQUERY (OPTION,VALUE)
UREAD	Allows alphanumeric input from the graphic terminal CALL UREAD (X,Y,DATA,COUNT,FLAG)
URECT	Draws a rectangle CALL URECT (X,Y)
URESET	Resets GSA variables to default conditions CALL URESET
UROTAT	Creates a user coordinate system at current position rotated as specified CALL ROTAT (ANGLE)
USAREA	Changes device boundaries to maintain a one to one aspect ratio with the current window boundaries CALL USAREA

<b>USAVE</b>	Saves all the variables of the Graphics Status Area CALL USAVE (ARRAY)
<b>USAXIS</b>	Draws a single axis in any of three coordinates CALL USAXIS (AXIS,XSTART,YSTART,ZSTART,DIST)
<b>USCALE</b>	Creates a user coordinate system at current position with specified scale CALL USCALE (SX,SY)
<b>USCATR</b>	Draws a scatter plot CALL USCATR (X,Y,PTS)
<b>USET</b>	Sets a graphics status area variable to a given value CALL USET (OPTION)
<b>USHOW</b>	Causes the named frame of graphical information to be made visible CALL USHOW (NAME)
<b>USPLIN</b>	Fits a cubic spline curve to the input data CALL USPLIN (X,Y,PTS,RETX,RETY,RETPTS)
<b>USTART</b>	Initializes the graphics status area CALL USTART
<b>USTRCT</b>	Defines the start of a graphic data structure CALL USTRCT (NAME)
<b>USTUD</b>	Returns limits of virtual and display surfaces CALL USTUD(S)
<b>USVPN</b>	Saves all pen-related variables of the graphics status area CALL USVPN (ARRAY)
<b>USVTR</b>	Saves coordinate system related variables in the graphics status area CALL USVTR (ARRAY)
<b>UTAXIS</b>	Draws a time series axis with appropriate alphameric and numeric labels CALL UTAXIS (BEGIN,PERIOD,YMIN,YMAX)
<b>UTERM</b>	Defines the end of a graphic data structure CALL UTERM (NAME)

UTILTY	Performs data structure utility functions CALL UTILTY (OPTION,VALUE)
UTSFIT	Fits an exponentially smoothed curve to time series data CALL UTSFIT (ARRAY,POINTS,FCST,ALPHA)
UVIEW	Defines position of viewer in relation to environment, and the direction of view. CALL UVIEW (XVIEW,YVIEW,ZVIEW,XSITE,YSITE,ZSITE)
UVWPLN	Defines the location of the view (projection) plane CALL UVWPLN (DISTAN)
UVWPRT	Defines the location of the view (projection) plane (obsolete, see UVWPLN) CALL UVWPRT (DISTAN)
UWAIT	Waits a given number of seconds CALL UWAIT (SECONDS)
UWHERE	Returns the coordinates of the current pen position in user units CALL UWHERE (X,Y)
UWINDO	Sets the virtual window boundaries CALL UWINDO (XMIN,XMAX,YMIN,YMAX)
UWLOOK	Adjusts both virtual window, and user display area to cover given portion of virtual space CALL UWLOOK (XMIN,YMAX,YMIN,YMAX)
UWRITE	Prints information, then restores pen to location on input CALL UWRITE (X,Y,DATA)
UWRIT1	Allows apphameric output at current position under one option CALL UWRIT1 (DATA,OPTION)
UZWNDO	Sets the hither/yon window boundaries for Z-clipping CALL UZWNDO (ZMIN,ZMAX)
U3AXIS	Creates a set of axes in three space CALL U3AXIS (XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)
U3CALL	Invokes an existing graphics data structure in three space. CALL U3CALL (X,Y,Z,SX,SY,SZ,RX,RY,RZ,NAME)



<b>U3CSYS</b>	Creates a new coordinate system in three space CALL U3CSYS (X,Y,Z,SX,SY,SZ,RX,RY,RZ)
<b>U3DRAW</b>	Draws a solid line in three space CALL U3DRAW(X,Y,Z)
<b>U3GRIN</b>	Gets 3-D coordinates and a character from terminal and returns them CALL U3GRIN (X,Y,Z,ICHAR)
<b>U3IMAG</b>	Applies general 3-D image transformations to 'retained' segments CALL U3IMAG (X,Y,Z,SX,SY,SZ,RX,RY,RZ,SEGNAM)
<b>U3LINE</b>	Draws a curve connecting three arrays of points (X,Y,Z) in three spaces CALL U3LINE (X,Y,Z,PTS)
<b>U3MOVE</b>	Moves pen invisibly in three space CALL U3MOVE (X,Y,Z)
<b>U3PEN</b>	Draws a line from current pen position to given 3-D coordinates CALL U3PEN (X,Y,Z)
<b>U3PEN1</b>	Sets one 'USET' option for this call only before executing 3-D pen movement CALL U3PEN1(X,Y,Z,OPTION)
<b>U3PLACE</b>	Applies 3-D translation image transformation to 'retained' segments CALL U3PLAC(X,Y,Z,SEGNAM)
<b>U3PLOT</b>	Draws a general purpose graph in three space CALL U3PLOT (X,Y,Z,CURVES,PTS,OPTS)
<b>U3PRNT</b>	Displays textual data at pen position in three space CALL U3PRNT (X,Y,Z,DATA)
<b>U3ROTA</b>	Creates a user coordinate system in three space at current pen position, rotated as specified CALL U3ROTA (RX,RY,RZ)
<b>U3SCAL</b>	Creates a user coordinate system in three space at current per position, scaled as specified CALL U3SCAL (SX,SY,SZ)
<b>U3STUD</b>	Gives current setting of the 3-D user display area and windows CALL U3STUD (ARRAY)

**U3WHER** Returns current pen position in current units in three space

CALL U3WHER (X,Y,Z)

**U3WNDO** Sets the virtual 3-D window boundaries

CALL U3WNDO (XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)

**U3WRIT** Displays textual information in three space, returns pen to original position

CALL U3WRIT (X,Y,Z,DATA)

**Subroutine UAIN***Interactive***FUNCTION:**

This routine obtains one character as entered to signify the completion of a graphics input operation. This is usually the result of a keyboard trike.

**CALLING SEQUENCE:****CALL UAIN (CHAR)**

Where

**CHAR** contains the character obtained from the terminal in Hollerith format.

**OPTIONS** which may apply:

'KEYBOARD', 'FUNCTIONKEY', 'CURSOR', 'JOYSTICK', 'BALL', 'MOUSE', 'LIGHTPEN',  
'TABLET'

**COMMENTS:**

The character requested will be obtained from the specified graphics input device. If no explicit USET call has been used to specify an input device (see UGRIN), the character will be obtained from the principal input device for the terminal being used.

The USET option 'BALL' refers to Trackball input.



### **Subroutine UALPHA**

#### **FUNCTION:**

This routine sets the terminal to alphanumeric mode so that character output will appear on the control device.

#### **CALLING SEQUENCE:**

**CALL UALPHA**

#### **OPTIONS which may apply**

No options apply.

#### **COMMENTS:**

This routine should be called whenever the user wants to insure that the terminal is an alphanumeric mode and to flush any buffered I/O (e.g., before FORTRAN I/O).

## **Subroutine UAOUT**

### **FUNCTION:**

This routine displays one character at the current beam position subject to the margins or windows which are currently set. The beam is positioned ready to output an adjacent character.

### **CALLING SEQUENCE:**

**CALL UAOUT (CHAR)**

Where

**CHAR** is an alphanumeric character expressed in FORTRAN Hollerith (left-justified, blank-filled) format.

### **OPTIONS which may apply:**

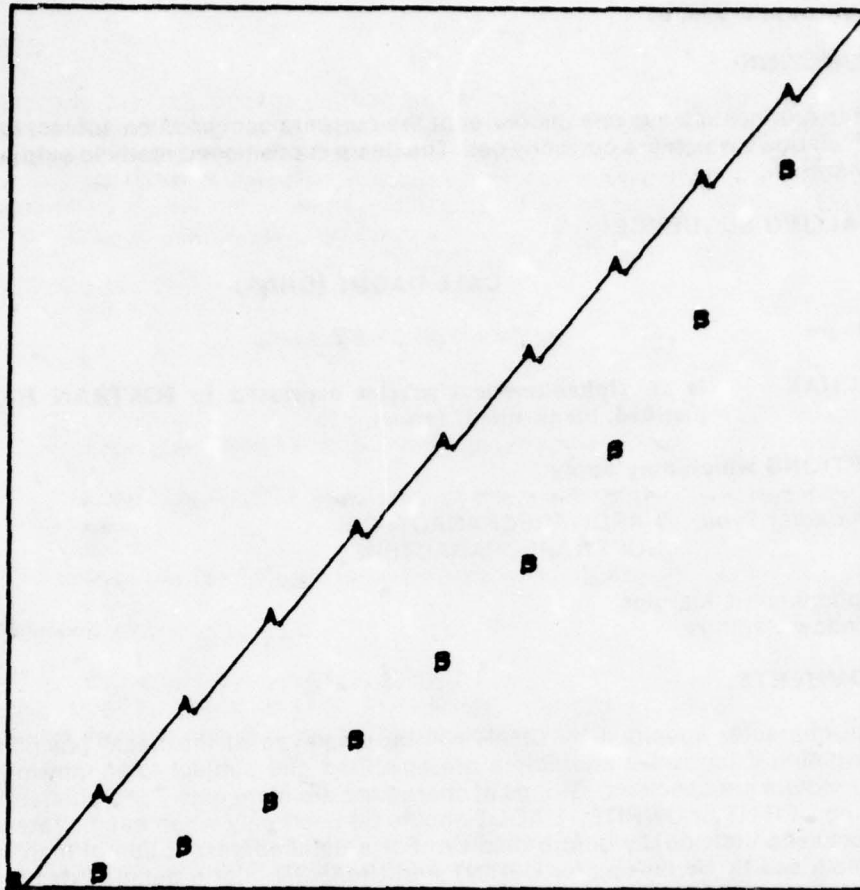
Character Type: 'HARDWARECHARACTERS'  
'SOFTWARECHARACTERS'

Alphanumeric Margins  
Window Settings

### **COMMENTS:**

The character specified by CHAR will be displayed at the beam position subject to *margin*ing if hardware characters are specified and subject to *window*ing if software characters are specified. Strings of characters are more easily and efficiently displayed using UPRINT or UWRITE. UAOUT should be used only when each character must be processed individually before display. For a detailed description of margining and its effects see the write-ups for UPRINT and UMARGN. For a detailed description of the concepts of windowing see the write-up for UWINDO.

### **Programming Notes:**



```

DIMENSION X(11),Y(11),Z(11)
DATA X/0.,10.,20.,30.,40.,50.,60.,70.,80.,90.,100./
DATA Y/0.,10.,20.,30.,40.,50.,60.,70.,80.,90.,100./
DATA Z/0.,1.,4.,9.,16.,25.,36.,49.,64.,81.,100./
CALL USTART
CALL UOUTLN
DO 1 I = 1, 11
  CALL UPEN (X(I),Y(I))
1 CALL UAOUT ('A')
  CALL USET ('NOLINE')
  CALL UMOVE (X(1),Z(1))
DO 2 I = 1, 11
  CALL UPEN (X(I),Z(I))
  CALL UAOUT ('B')
2 CONTINUE
CALL UEND
STOP
END

```



## **Subroutine UAPEND**

### **FUNCTION:**

This routine copies characters from the input Hollerith character string placing them in the output character buffer. When the designated number of characters have been moved, the GCS string terminator is appended to the end of the string in the output buffer.

### **CALLING SEQUENCE:**

**CALL UAPEND(COUNT,DATAIN,DATOUT)**

Where

**COUNT** is the number of characters in the input Hollerith character string.

**DATAIN** is the input Hollerith character string.

**DATOUT** is available or array large enough to hold COUNT+1 characters

### **OPTIONS which may apply:**

UPSET('TERMINATOR',CHAR)

### **COMMENTS:**

This routine is useful if a character string is being obtained from an input source and the GCS string terminator has not been included at the end of the string. It is assumed that enough room is available in the output string buffer area to insert the GCS string terminator character.

DATAIN, and DATOUT need not be distinct.

### **Programming Notes:**

## **Subroutine UARC**

### **FUNCTION:**

This subroutine draws a part of a circle; i.e., an arc with a given center, radius, and angular span starting at the current beam or pen position. The arc will be either treated as a curvilinear line segment with the current line option (see UPEN for description of line options) or treated as a collection of straight line segments, each of which is affected by the current line option. Which method is used to draw the arc will depend on whether 'CONTINUOUS' or 'SEGMENTED' has been set.

### **CALLING SEQUENCE:**

**CALL UARC (X,Y,ANGLE)**

Where

**X** is the X or RADIUS of the center of the circle in current user units.  
**Y** is the Y or THETA coordinate of the center of the circle in current user units.  
**ANGLE** is the number of angular units to be subtended by the arc.

### **OPTIONS which may apply:**

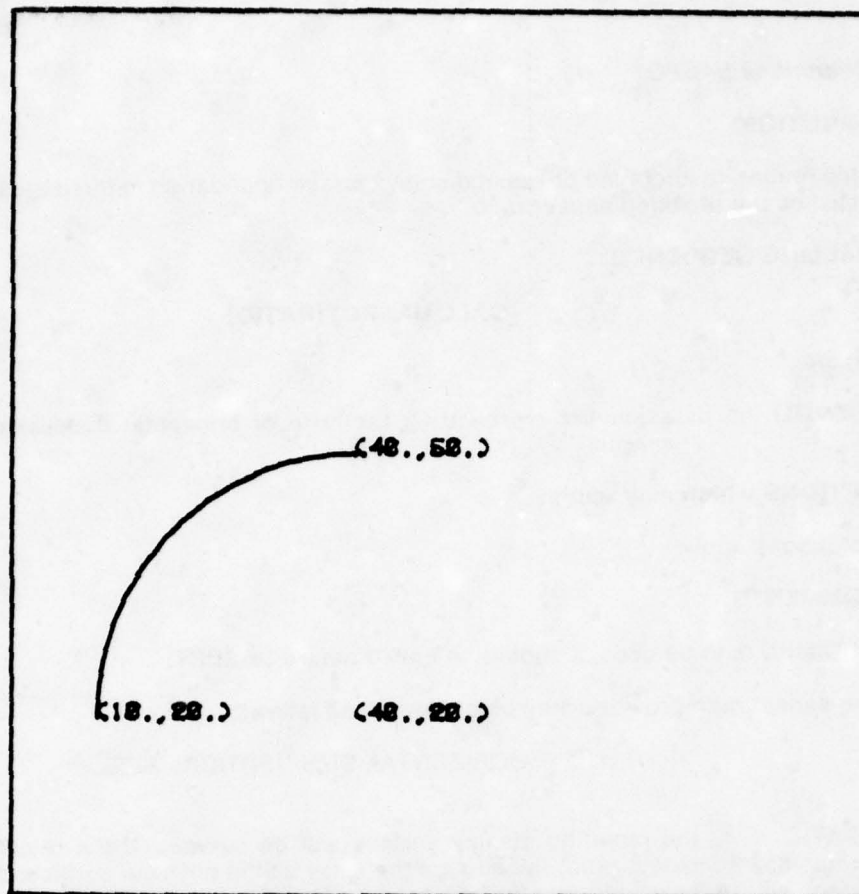
Arc Type: 'CONTINUOUS' or 'SEGMENTED'  
Angle Units: 'DEGREES', 'RADIANS', 'PIRADIANS', 'GRADS' or 'MILS'  
Line Types: Any line option available under UPEN

### **COMMENTS:**

The arc will be drawn counterclockwise if ANGLE is positive or clockwise if ANGLE is negative. The arc will extend from the beam or pen position upon invoking UARC for the angle specified in current angular units (Degrees, Radians, or *Piradians*).

*In three dimensional applications, the arc produced by a call to UARC will lie in the current X-Y plane.*

### **Programming Notes:**



```
CALL USTART
CALL UOUTLN
CALL UPEN1 (48.0, 58.0, 'NCOORDINATES')
CALL UMOVE (48.0, 58.0)
CALL UARC (48.0, 28.0, 90.0)
CALL UWHERE (X, Y)
CALL UPEN1 (X, Y, 'NCOORDINATES')
CALL UPEN1 (48.0, 28.0, 'NCOORDINATES')
CALL UEND
STOP
END
```



**FUNCTION:**

This routine restricts the physical display surface boundaries to the largest area which satisfies the provided aspect ratio.

**CALLING SEQUENCE:**

**CALL UASPCT(RATIO)**

Where

**RATIO** is a number representing the ratio of horizontal dimension to vertical dimension.

**OPTIONS which may apply:**

No options apply

**COMMENTS:**

If UDIMEN is to be used, it should be called before UASPCT.

The aspect ratio provided may be computed as follows:

$$\text{RATIO} = \text{HORIZONTAL SIZE} / \text{VERTICAL SIZE}$$

If  $\text{RATIO} = 1.$ , the resulting display surface will be square. If the resulting display surface has an aspect ratio which is not the same as the physical surface, the resulting surface will be lower-left justified on the physical display. If  $\text{RATIO} = 0.$ , the entire display surface will be used. The default aspect ratio is the entire display surface.

**Programming Notes:**

### **Subroutine UAVERG**

#### **FUNCTION:**

To compute a moving average forecast for time series data.

#### **CALLING SEQUENCE:**

**CALL UAVERG (ARRAY,POINTS,FCST,TIME)**

Where

**ARRAY** is a real array of size **POINTS** which is the input time series data.

**POINTS** is the number of input points.

**FCST** is a real array of size **POINTS-TIME+1** which is the time series moving average forecast for periods **TIME+1** to **POINTS+1**

**TIME** is the number of previous periods to be used for each average.

#### **OPTIONS which may apply:**

No options apply.

#### **COMMENTS:**

The number of input data points must be greater than one. The number of previous periods which are used to compute the moving average must be greater than zero and not greater than the number of input data points.

Note that this routine represents one of the basic tools for time series analysis. The user is urged to consult a qualified reference on time series analysis for a complete description of the methods involved in this routine. Other techniques such as base series correction, cyclic analysis, and exponential smoothing are necessary to synthesize a meaningful, composite forecast.

#### **Programming Notes:**

## Subroutine UAXIS

### FUNCTION:

This routine creates axes of the type specified by the user. Also provided are numeric scaling and labeling and alphanumeric labeling. The axes are created in the UDAREA which is set prior to entering UAXIS. The user window may also be modified to reflect the actual scaling used. Upon return from UAXIS, the user window will be set to the scaling used to draw the axes. The device area will be reduced if 'EDGEAXES' are specified to accommodate the labeling requested. If 'ZEROAXES' or 'PENAXES' are specified, the device area will not be changed since the labeling will appear in the device area setting.

### CALLING SEQUENCE:

**CALL UAXIS (XMIN,XMAX,YMIN,YMAX)**

Where

<b>XMIN</b>	is the minimum X-value to be contained on the axis in virtual units.
<b>XMAX</b>	is the maximum X-value to be contained on the axis in virtual units.
<b>YMIN</b>	is the minimum Y-value to be contained on the axis in virtual units.
<b>YMAX</b>	is the maximum Y-value to be contained on the axis in virtual units.

### OPTIONS which may apply:

Axis Existence:	'NOAXES', 'XAXIS', 'YAXIS', 'XYAXIS'
Axis Format:	'PLAINAXES', 'TICAXES', 'GRIDAXES'
X-Axis Labeling:	'XNUMERIC', 'XALPHALABEL', 'XBOTH', 'NOXLABEL'
Y-Axis Labeling:	'YNUMERIC', 'YALPHALABEL', 'YBOTH', 'NOYLABEL'
Axis Positioning:	'ZEROAXES', 'XZEROYEDGE', 'YEDGEXZERO', 'XEDGEYZERO', 'YZEROXEDGE', 'EDGEAXES', 'PENAXES'

### Numeric Label

Format:	'IFORMAT', 'GFORMAT', 'BESTFORMAT'
Scaling Type:	'AUTOSCALE', 'FULLSCALE', 'OWNSCALE'
X-Axis Type:	'LINXAXIS', 'LOGXAXIS', 'LNYAXIS'
Y-Axis Type:	'LINYAXIS', 'LOGYAXIS', 'LNYAXIS'
Coordinate Type:	'POLAR', 'RECTANGULAR', 'LOGARITHMIC'

### COMMENTS:

UAXIS is a general purpose axis drawing and/or plot labeling routine. The options indicated above are independent of each other type designations so that several different options might appropriately be chosen. The UAXIS subroutine is also used by the high-level graphing routines (see UPLOT) to create their axes.

Each of the USET options which effect the UAXIS routine is described below. These same options apply when subroutine UPLOT or subroutine UPLOT1 is called.

### Programming Notes:



#### **Axis Existence:**

- |               |  |
|---------------|--|
| <b>NOAXES</b> | — Neither the X-axis nor the Y-axis will be displayed. |
| <b>XAXIS</b>  | — Only the X-axis will be displayed                    |
| <b>YAXIS</b>  | — Only the Y-axis will be displayed                    |
| <b>XYAXES</b> | — both the X- and Y-axis will be displayed             |

#### **Axis Format:**

- |                  |   |
|------------------|---|
| <b>PLAINAXES</b> | — The axis specified will appear as a solid line.   |
| <b>TICAXES</b>   | — The axis specified will appear as a ticced line. The tic intervals will be determined from the current setting of the UPSET parameters TICX and TICY which should be specified in the same units to be used in plotting the functions. If a tic interval of zero is specified, a suitable tic interval will be chosen by UAXIS.   |
| <b>GRIDAXES</b>  | — A grid will be formed in the display area. The interval between grid lines is determined in the same way as for TICAXES. In the case of log axes, the tic interval refers to the cycle interval. In the case of polar axes, the tic interval refers to the radial distance between the concentric circles of the grid. This radial distance is obtained from the value of TICX. |

#### **Axis Positioning:**

The user is expected to establish the UDAREA he desires before calling UAXIS. For all the options indicated below, no part of the axes or the associated labels will appear outside the UDAREA which is set prior to entry to UAXIS.

- |                                 |  |
|---------------------------------|--|
| <b>EDGEAXES</b>                 | — The room needed for the 'axes' specified labeling is reserved from the UDAREA set on input and a new UDAREA is established which describes the remaining plotting area. The axes will be drawn along the edge of this new UDAREA.  |
| <b>ZEROAXES</b>                 | — If the zero point falls between the minimum and maximum input values for either X or Y, the axes are drawn along the zero value from border to border of the UDAREA. Numeric labels will appear directly below the X-axis and to the left of the Y-axis. Alphanumeric X-labels will appear along the bottom edge of the UDAREA and the alphanumeric Y-labels will appear along the left edge of the UDAREA. If the zero point for either axes is not between the minimum input values, the axis will default to EDGEAXIS format. |
| <b>PENAXES</b>                  | — The intersection of the axes is defined to be at the beam position upon entry to UAXIS. If the beam position is outside the range of values furnished as input, the axis which is outside the range will default to EDGEAXIS format.   |
| <b>YEDGEZERO<br/>XZEROYEDGE</b> | — The X-axis will be in ZEROAXIS format and the Y-axis will be in EDGEAXIS format.   |
| <b>XEDGEZERO<br/>YZEROXEDGE</b> | — The X-axis will be EDGEAXIS format and Y-axis will be in ZEROAXIS format.  |

**X-Axis Type:**

- LINXAXIS — The X-axis is in linear Cartesian format.
- LOGXAXIS — The X-axis is the common (base 10) logarithmic format.
- LNAXIS — The X-axis is in natural (base e) logarithmic format.

**Y-Axis Type:**

- LINYAXIS — The Y-axis is in linear Cartesian format.
- LOGYAXIS — The Y-axis is in common (base 10) logarithmic format.
- LNAYAXIS — The Y-axis is in natural (base e) logarithmic format.

**Coordinate Type:**

- RECTANGULAR — The input coordinates are rectangular. Logarithmic or linear axes can be specified.
- POLAR — Input coordinates are polar. Only polar axes will be created by UAXIS.
- LOGARITHMIC — Input coordinates are rectangular logarithmic. Logarithmic axes will be generated as specified by the axis type options.

**X-Axis Labeling:**

- NOXLABEL — No labeling should appear along the X-axis.
- XNUMERIC — Numeric labels should be created along the X-axis at the tic-intervals specified by TICX (see Axis Format above). The values for the numeric labels will be derived as specified in the scaling options described below.
- XALPHANUMERIC — Alphanumeric labels will appear in the location specified in the Axis Position Option (see above). The contents of the label can be specified by calling UPSET with arguments 'XLABEL' and a GCS character string as the value (for a description of GCS character strings see UPRINT).
- XBOTHLABELS — Both the alphanumeric and numeric labels will appear along the X-axis.

**Y-Axis Labeling:**

- NOYLABEL — No labeling should appear along the Y-axis.
- YNUMERIC — labels should be created along the Y-axis at the tic intervals specified by TICY (see Axis Format above). The values for the numeric labels will be derived as specified in the scaling options described below.
- YALPHANUMERIC — Alphanumeric labels will appear in the location specified by the Axis Positioning Option (see above). The contents of the label can be specified by calling UPSET with

arguments 'YLABEL' and a GCS character string as the value (for a description of GCS character strings see UPRINT).

**YBOTHLABELS**

- Both the alphanumeric and numeric labels will appear along the Y-axis.

**Scaling Type:**

**AUTOSCALE**

- The range of values to be plotted is examined and scaling occurs so that 'nice' numbers will appear at the tic marks. A slight expansion of the range may be necessitated to incorporate this feature. The UWINDO is set to the expanded range as identified.

**FULLSCALE**

- The full range of values specified in the input arguments are examined. If zero is not within the range of X values, the appropriate limit is extended to zero. The UWINDO is set to this resulting range.

**OWNSCALE**

- The UWINDO setting upon entry to UAXIS is used to determine the scale limits. No change is made to the UWINDO and a zero value is not forced to the X -scale. 'Nice' numbers may not necessarily appear as numeric labels.

**Numeric Label Format:**

**IFORMAT**

- All numeric labels will appear as integers. Truncation of fractional values may occur.

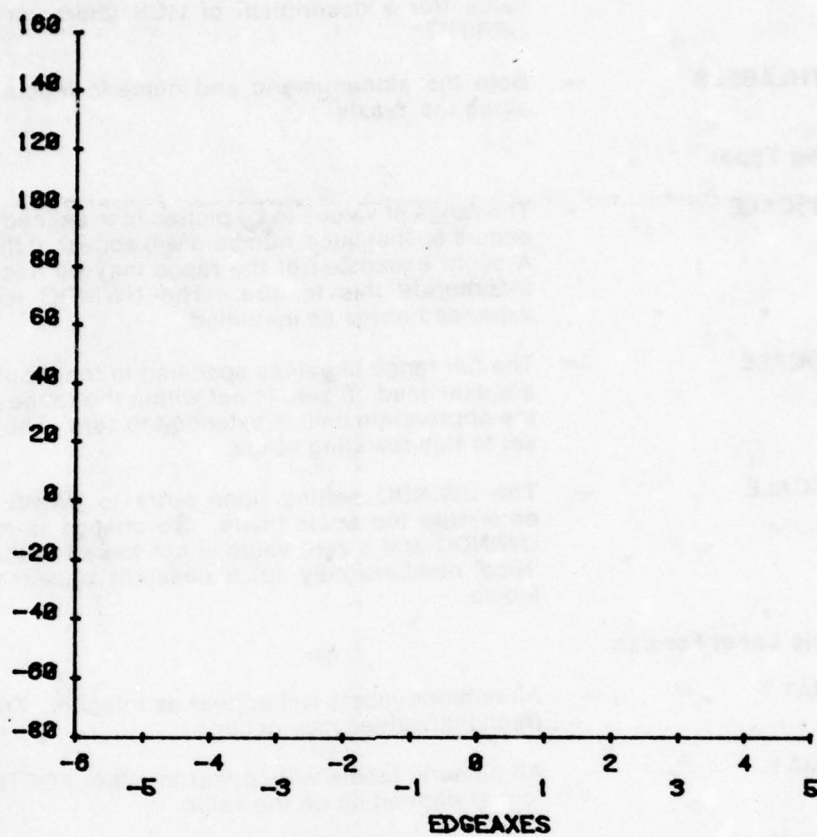
**GFORMAT**

- All numeric labels will appear in either FORTRAN F or E format depending on the value.

**BESTFORMAT**

- The numeric labels will appear in the format which is most appropriate to the value. If the value is an integer, it will appear as IFORMAT. Otherwise, it will appear as GFORMAT.

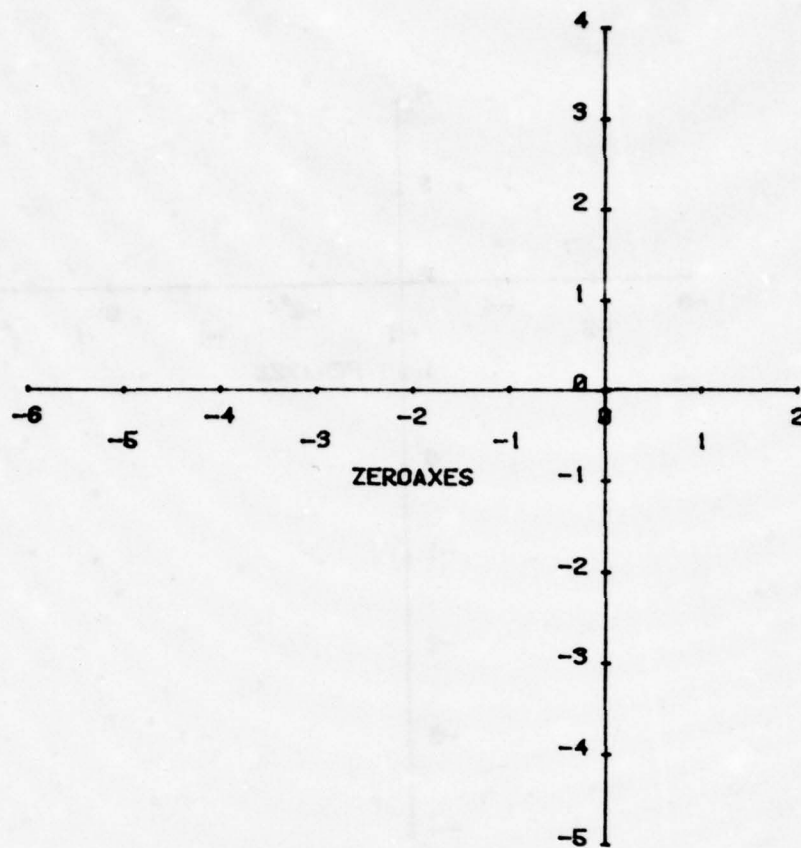




```

CALL USTART
CALL UPSET ('TERMINATOR',',','')
CALL USET ('XBOTHLABELS')
CALL UPSET ('XLABEL',',',EDGEAXES,')
CALL UAXIS (-5.32,4.89,-89.4,156.459)
CALL UEND
STOP
END

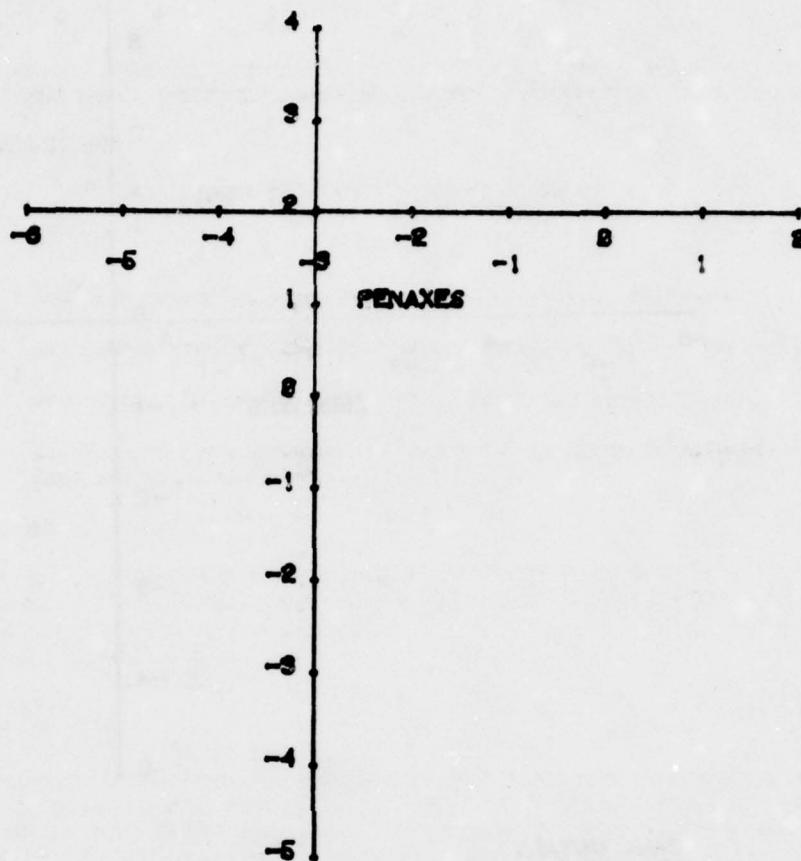
```



```

CALL USTART
CALL UPSET ('TERMINATOR','(',')')
CALL USET ('ZEROAXES')
CALL USET ('XBOTHLABELS')
CALL UPSET ('XLABEL','ZEROAXES,')
CALL UAXIS (-6.,2.,-5.,4.)
CALL UEND
STOP
END

```

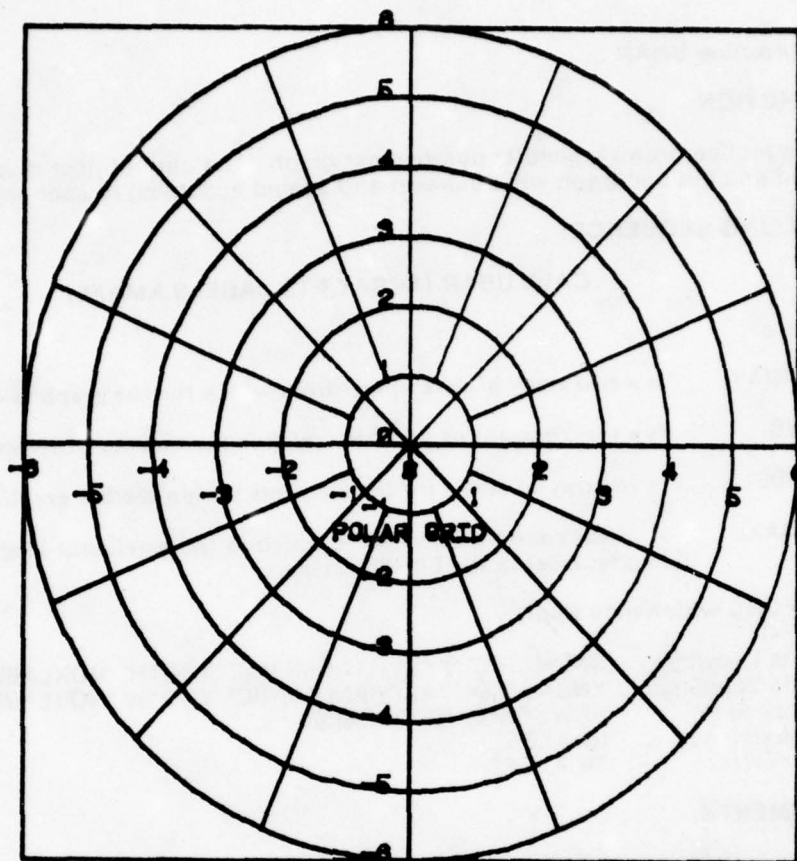


```

CALL USTART
CALL UPSET ('TERMINATOR','')
CALL USET ('PENAXES')
CALL USET ('XBOTHLABELS')
CALL UPSET ('XLABEL','PENAXES,')
CALL UMOVE (-3.,2.)
CALL UAXIS (-6.,2.,-5.,4.)
CALL UEND
STOP
END

```





```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UPSET ('XLABEL','POLAR GRID,')
CALL USET ('POLAR')
CALL USET ('GRID')
CALL USET ('XBOT')
CALL USET ('ZEROAXIS')
CALL UAXIS (-6.,6.,0.,360.)
CALL UEND
STOP
END

```

## **Subroutine UBAR**

### **FUNCTION:**

This routine draws a general purpose bar graph. The current user display area will be used and the bar graph will be drawn and scaled according to user specifications.

### **CALLING SEQUENCE:**

**CALL UBAR (ARRAY,PTS,LABELS,XMAXL)**

Where

**ARRAY** is a real array of data points from which the bar graph is drawn.  
**PTS** is a real constant or variable which is the number of elements in **ARRAY**.  
**LABELS** is an array of labels for the elements within the bar graph.  
**XMAXL** is a real constant or variable which is the maximum length of a label or longest label in the **LABELS** array.

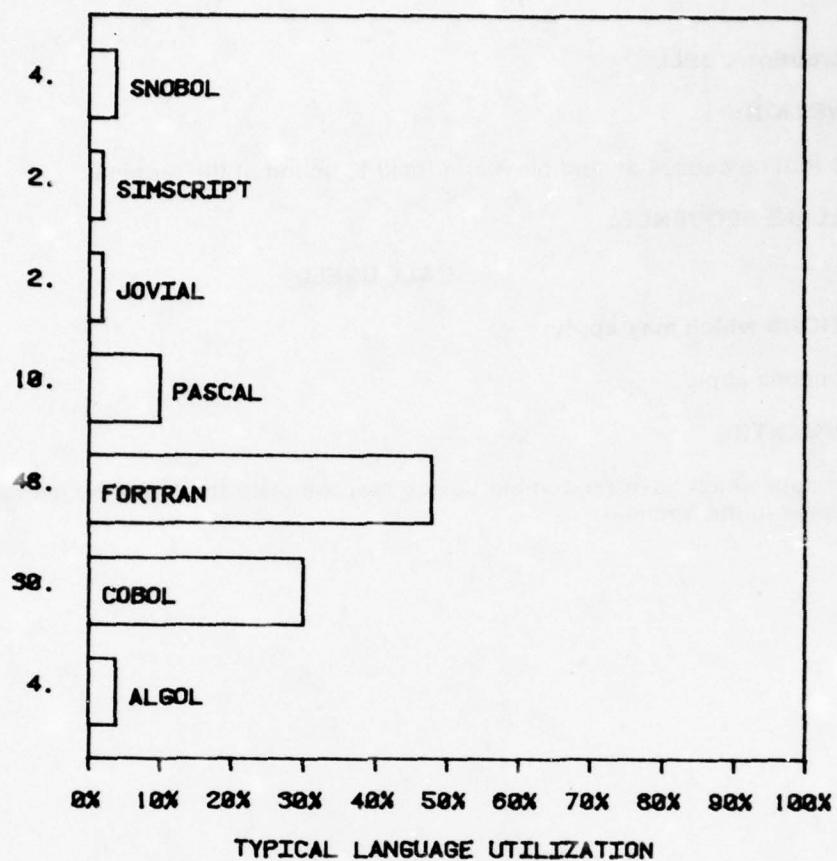
### **OPTIONS which may apply:**

X-Axis Labeling: 'XNUMERIC' 'XALPHANUMERIC' 'XBOTH' 'NOXLABEL'  
Y-Axis Labeling: 'YNUMERIC' 'YALPHANUMERIC' 'YBOTH' 'NOYLABEL'  
Display Area: 'NEWSCALE' 'OLDSCALE'  
Axis Existence: 'NOAXES'  
Axis Format: 'TICAXES'

### **COMMENTS:**

UBAR will place the labels from the label array on the individual bars starting from the bottom up. All input data is normalized so that each input value will appear as a percentage of the total of the data array. The specification of anything other than 'TICAXES' will cause plain axes to be drawn. The specification of 'NOAXES' will cause suppression of graphic output so that only the labels and values will appear.

### **Programming Notes:**



```

DIMENSION DATA(7)
CHARACTER LABELS*10(7)
DATA DATA/4.,38.,48.,10.,2.,2.,4./
DATA LABELS/'ALGOL','COBOL','FORTRAN','PASCAL','JOVIAL',
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('XBOOTHLABELS')
CALL UPSET ('XLABEL','TYPICAL LANGUAGE UTILIZATION,')
CALL UBAR (DATA,7.,LABELS,10.)
CALL UEND
STOP
END

```



**Subroutine UBELL**

*Interactive*

**FUNCTION:**

This routine causes an audible alarm (bell) to sound at the terminal.

**CALLING SEQUENCE:**

**CALL UBELL**

**OPTIONS which may apply:**

No options apply.

**COMMENTS:**

Terminals which have no audible device may simulate the alarm by sending a visual message to the terminal.

**FUNCTION:**

This routine invokes an already constructed data structure. The request coordinate system transformation is established and then each element of the structure is executed.

**CALLING SEQUENCE:**

**CALL UCALL(X,Y,SX,SY,R,NAME)**

Where

- X,Y** are the coordinates of the origin of the structure coordinate system in current coordinate system units.
- SX, SY** are the scale factors of the structure coordinate system.
- R** is the rotation of the structure coordinate system in relation to the current reference system.
- NAME** is the eight character name of the structure being invoked. This structure must be defined in the current 'LIBRARY' file.

**COMMENTS:**

The current coordinate system is saved prior to applying the structure coordinate transformation. Up to three levels of recursive calls to UCALL can be handled.

This subroutine can be used in a 3-D environment if the transformation so specified is to be applied only in the XY plane of the current coordinate system.

**Programming Notes:**

## **Subroutine UCHART**

### **FUNCTION:**

This routine draws a grouped bar chart for multi-valued data. The current user display area will be used and the chart will be drawn and scaled according to user specifications.

### **CALLING SEQUENCE:**

**CALL UCHART(ARRAY, GROUPS, BARS, LABELS, YMAXL)**

Where

<b>ARRAY</b>	is a real array of data points from which the chart is drawn
<b>GROUPS</b>	is a real constant or variable which is the number of groups
<b>BARS</b>	is a real constant or variable which is the number of bar graphs in each group
<b>LABELS</b>	is an array of labels for each group
<b>YMAXL</b>	is a real constant or variable which is the maximum length of a label or the length of the longest label in the array LABELS

### **OPTIONS which may apply:**

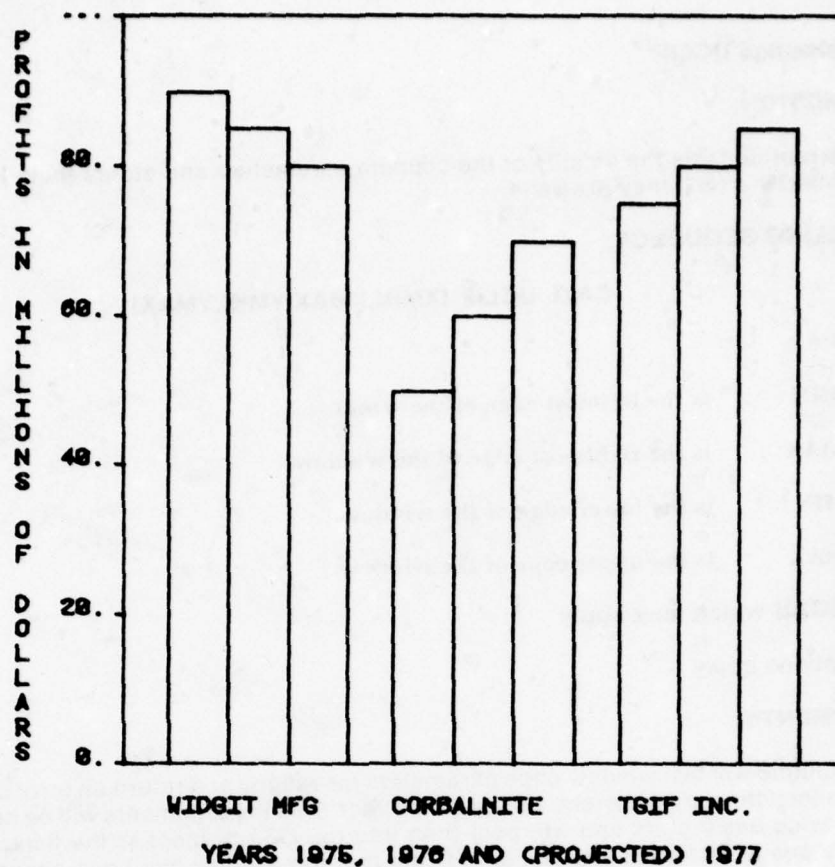
X-Axis Labeling:	'XNUMERIC', 'XALPHABETIC', 'XBOTH', 'NOXLABEL'
Y-Axis Labeling:	'YNUMERIC', 'YALPHABETIC', 'YBOTH', 'NOYLABEL'
Display Area:	'NEWSCALE', 'OLDSCALE'
Axis Existence:	'NOAXES'
Axis Format	'TICAXES'

### **COMMENTS:**

The specification of anything other than 'TICAXES' will cause plain axes to be drawn. The specification of 'NOAXES' will cause suppression of graphic output so that only the labels and values will appear. The X-axis and Y-axis labeling options will not affect the output of the group labels. If group labels are not desired, then YMAXL should be set to zero.

### **Programming Notes:**





```

DIMENSION YEARS(9)
CHARACTER LABELS*18(3)
DATA YEARS/99., 85., 80., 50., 60., 70., 75., 80., 85./
DATA LABELS/'WIDGIT MFG,', 'CORBALNITE,', 'TGIF INC.,'/
CALL USTART
CALL UPSET ('TERMINATOR', ',')
CALL USET ('XALPHABETIC')
CALL UPSET ('XLABEL', 'YEARS 1975, 1976 AND (PROJECTED) 1977,')
CALL USET ('YBOTHLABELS')
CALL UPSET ('YLABELS', 'PROFITS IN MILLIONS OF DOLLARS,')
CALL UCHART (YEARS, 3., 3., LABELS, 18.)
CALL UEND
STOP
END

```

### **Subroutine UCLIP**

#### **FUNCTION:**

This routine tests the validity of the clipping parameters and stores them into the GCS COMMON area if they are valid.

#### **CALLING SEQUENCE:**

**CALL UCLIP (XMIN,XMAX,YMIN,YMAX)**

Where

**XMIN** is the leftmost edge of the window.  
**XMAX** is the rightmost edge of the window.  
**YMIN** is the lower edge of the window.  
**YMAX** is the upper edge of the window.

#### **OPTIONS which may apply**

No options apply

#### **COMMENTS:**

This routine will test window edge parameters for validity and return an error condition to the caller if they are incorrect. If they are correct, then the arguments will be converted to the device basic units and will post then into the GSA subject to the limitation of the device space. A flag is also set to indicate that the window has been changed.

#### **Programming Notes:**

**FUNCTION:**

This routine causes the currently open segment/frame to cease to be open.

**CALLING SEQUENCE:**

**CALL UCLOSE(SEGID)**

Where

**SEGID** is the segment identifier of the currently open segment (see UOPEN for description of SEGID).

**OPTIONS which may apply:**

Segment/Frame Identifier Mode: 'FNAME', 'FNUMBER'

**COMMENTS:**

The currently open segment is closed irrelevant of the type of segment or the existence of image transformations. The closed frame will replace an existing segment with the same identifier thus insuring that a complete image is always visible.

An error will be generated if SEGID does not agree with that of the currently open segment. However, the segment will still be closed.

**Programming Notes:**



## Subroutine UCONIC

### FUNCTION:

This routine allows the user to draw a generalized conic section having a given focus, directrix, eccentricity, and starting and ending sweep positions. The conic sections that may be drawn are the circular arc, ellipse, parabola, hyperbola, and, if the distance between the focus and the directrix is zero, point.

### CALLING SEQUENCE:

CALL UCONIC (X,Y,P,E,TH1,TH2)

Where

- X is the X- or RADIUS coordinate of the focus of the conic section.
- Y is the Y- or THETA coordinate of the focus of the conic section.
- P is the distance from the focus to the directrix.
- E is the eccentricity.
- TH1 is the initial sweep position in current angular units.
- TH2 is the terminal sweep position in current angular units.

### OPTIONS which may apply:

'CONTINUOUS', 'SEGMENTED' (see UARC write up)

Line options (see UPEN write up)

'RADIANS', 'DEGREES', 'PIRADIANS', 'GRADS' or 'MILS' (see UARC write up)

Pen Coordinate Options (see UPEN write up)

UPSET Option: 'ORIENTATION'

### COMMENTS:

The basis for UCONIC is the generalized conic equation:

$$R = (E \cdot P) / (1 - E \cdot \cos(\text{THETA}))$$

By suitably modifying the values of the parameters P and E, all types of conic sections can be created. The following describes the effects induced by different values of P of E.

- PGT0 the focus is to the right (below the directrix).
- PGT0 the focus is to the left (above the directrix).
- P = 0 the conic will be a point at X,Y.
- E = 0 the conic is a circular arc with center at X,Y and RADIUS P/2 subtending the angular range from the TH1 to TH2.
- OGT/E/GT1 the conic is an ellipse.
- /E=-1 the conic is a parabola.
- /E/GT1 the conic is a hyperbola.

EGTO            the conic is oriented along the X-axis

EGTO            the conic is oriented along the Y-axis

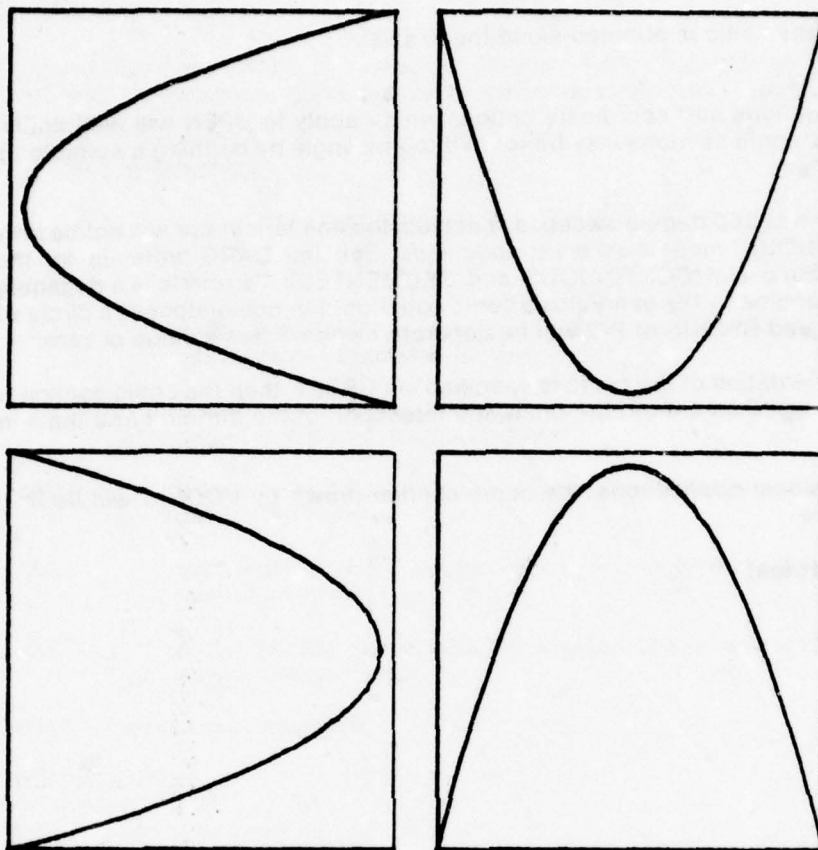
All of the line options and coordinate options which apply to UPEN will also apply to UCONIC. Thus, conic sections may be rotated to any angle by defining a suitable user coordinate system.

If a conic section of 360 degree sweep is specified, the line terminator will not be drawn when 'CONTINUOUS' mode has been specified. See the UARC write up for more information on the use of 'CONTINUOUS' and 'SEGMENTED'. The circle is a degenerate case not fully handled by the generalized conic equation. For completeness a circle with arbitrarily assigned RADIUS of P/2 will be generated when E has a value of zero.

If an angle of orientation of the conic is specified via UPSET, then the conic section will be oriented as specified around the point of intersection of the directrix and the semi-major axis.

*In three dimensional applications, the conic section drawn by UCONIC will lie in the current XY plane.*

**Programming Notes:**



```

DIMENSION X(4),Y(4),P(4),E(4)
DATA INDEX,Y0,X,Y,P,E/0,5.73,10.,50.,90.,50.,50.,10.,50.,90.,
& 13.,-13.,-13.,13.,1.,-1.,1.,-1./
CALL USTART
DO 1 I = 1, 2
  X0 = -1.62
  Y0 = Y0 - 2.66
  DO 1 J = 1, 2
    X0 = X0 + 2.66
    INDEX = INDEX + 1
    CALL UDAREA (X0,(X0+2.57),Y0,(Y0+2.57))
    CALL UOUTLN
    CALL UCONIC (X(INDEX),Y(INDEX),P(INDEX),E(INDEX),0.0,360.0)
1 CONTINUE
CALL UEND
STOP
END

```



## Subroutine UCONTR

### FUNCTION:

This routine draws a set of user specified contours on a set of data  $Z = Z(X,Y)$ , where  $Z$  is defined on a regular rectangular array.

### CALLING SEQUENCE:

**CALL UCONTR(Z,X,Y,A,FX,FY,CURVE,FN)**

Where

- Z** is a real array having dimensions IFIX(FX) by IFIX(FY) containing the function values.
- X** is a real array of length IFIX(FX) containing the X axis values
- Y** is a real array of length IFIX(FY) containing the Y axis values
- FX** floating point representation of the length of the X array, also the leading dimension of the Z array in floating point format
- FY** floating point representation of the length of the Y array, also the second dimension of Z in floating point format
- CURVE** array containing the values of the contours which are to be drawn
- FN** number of contours to be drawn. Also, the length (in floating point format) of CURVE
- A** working array of same size and dimensions as Z.

### OPTIONS which may apply:

Coordinate Type Options:	'RECTANGULAR', 'POLAR', 'LOGARITHMIC', UAXIS Options, (see UAXIS write up)
Scaling Type Options:	'AUTOSCALE', 'FULLSCALE', 'OWNSCALE'
Scale Availability Options:	'NEWSCALE', 'OLDSCALE'
Pen Options:	See UPEN write up

### COMMENTS:

The algorithm used in UCONTR is based on linear interpolation between the grid points.

### Programming Notes:

## **Subroutine UCOSYS**

### **FUNCTION:**

The specified user coordinate system is created using the position, scale, and rotation specified in the arguments, and plotting in the new coordinate system is automatically activated.

### **CALLING SEQUENCE:**

**CALL UCOSYS (X,Y,SCLX,SCLY,ANGLE)**

Where

- |              |  |
|--------------|--|
| <b>X</b>     | is the X- or RADIUS coordinate (in the current coordinate system) of the origin of the coordinate system.    |
| <b>Y</b>     | is the Y- or THETA coordinate (in the current coordinate system) of the origin of the new coordinate system. |
| <b>SCLX</b>  | is the ratio of the unit length in the new X direction to the unit length in the current X direction.        |
| <b>SCLY</b>  | is the ratio of a unit length in the new Y direction to the unit length in the current Y direction.          |
| <b>ANGLE</b> | is the rotation factor which is applied to the new coordinate system in current angular units.               |

### **OPTIONS which may apply:**

'WORKINGAXIS'  
'REFERENCEAXIS'  
UPSET option 'ZVALUE'

### **COMMENTS:**

The execution of this subroutine causes the creation of a new origin (0,0) at the specified position with respect to the current origin. The new coordinate system is rotated about the new origin by the specified angle and has unit scales in the X and Y direction as specified. If GCS is in 'WORKINGAXIS' mode, the coordinate system is created on a temporary basis. The coordinate system composition has no effect on subsequent coordinate system composition. If GCS is in 'REFERENCEAXIS' mode, the coordinate system composition is based upon the current coordinate system. All coordinate system or 'SYSTEM' coordinate system. At any time the user can ensure that his secondary coordinate system is composed with respect to the system coordinate system by the use of the following GCS subroutine call:

**CALL USET('SYSTEM')**

In addition, the above call may be used at any time to ignore the current user coordinate system. To restore the current user coordinate system, the following GCS subroutine call is used:

**CALL USET ('USERAXIS')**

In a sense, a user always has two user coordinate systems, a permanent or reference system, and a temporary or working system. In 'WORKINGAXIS' mode, the working system is always 'ahead of' the reference system in a mathematical sense. In

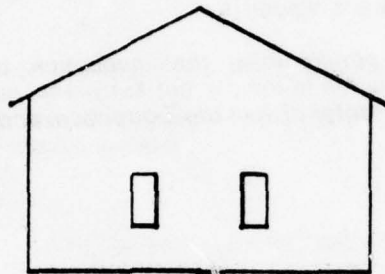
'REFERENCEAXIS' mode, the working system is mathematically identical to the reference system. thus no new coordinate system is invoked in switching from reference system mode to working system mode. However, by switching in the opposite direction, the last reference system becomes the current system.

Considerable care should be exercised when utilizing the 'REFERENCEAXIS' of cumulative mode, especially in conjunction with non-uniform axis scaling. A coordinate system which is superimposed upon a nonuniform coordinate system and rotated by a non-zero angle will contain a skew factor resulting in axes which are not normal to each other. Graphic output drawn in such a system, although mathematically correct, may be difficult to understand and visualize.

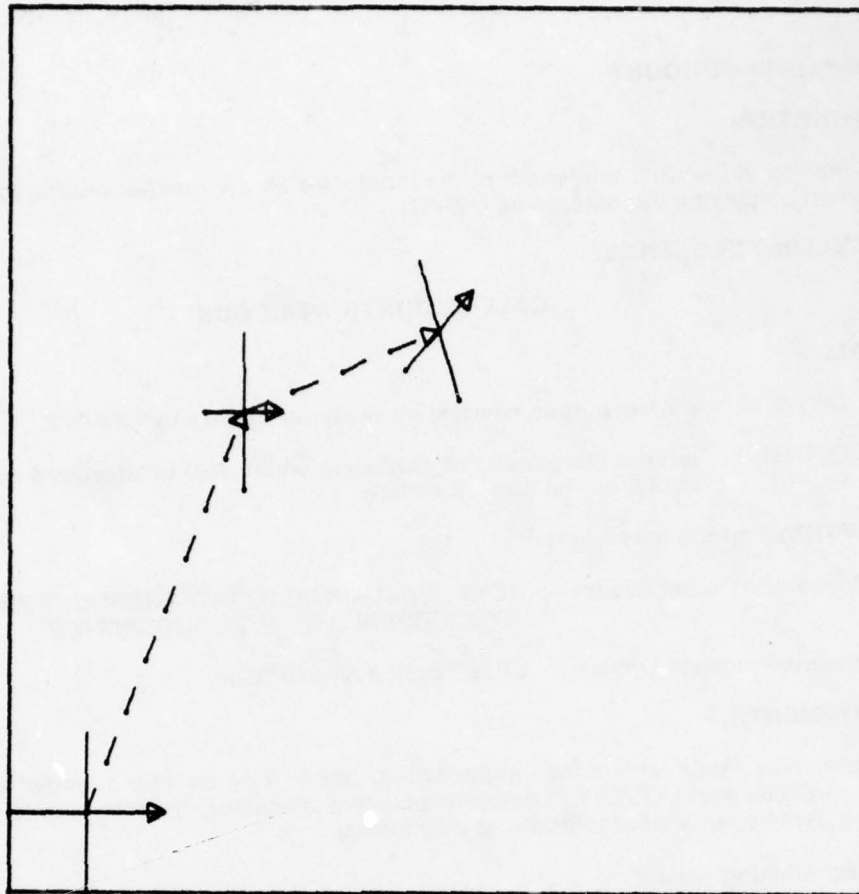
*In three dimension applications, the invocation of UCOSYS performs a three dimensional transformation in the current X-Y plane, specified by the UPSET parameter 'ZVALUE', with a scale factor of 1. in the Z direction, and rotation factors of 0. around the X and Y axis.*

**Programming Notes:**





```
CALL USTART
CALL UOUTLN
CALL UCOSYS (50.,50.,1.,1.,0.)
I = 1
100 CALL UMOVE (0.,-10.)
CALL UPEN (20.,-10.)
CALL UPEN (20.,10.)
CALL UMOVE (22.,0.)
CALL UPEN (0.,20.)
CALL UMOVE (5.,-5.)
CALL URECT (8.,1.)
IF (I.EQ. 2) GO TO 900
CALL UCOSYS (50.,50.,-1.,1.,0.)
I = 2
GO TO 100
900 CONTINUE
CALL UEND
STOP
END
```



```

CALL USTART
CALL UWINDO (-1.,10.,-1.,10.)
CALL UOUTLN
CALL AXIS
CALL UMOVE (0.,0.)
CALL USET ('REFERENCE')
CALL UCOSYS (2.,5.,.5,1.,0.)
CALL UPEN1 (0.,0., 'DARROW')
CALL AXIS
CALL UMOVE (0.,0.)
CALL UCOSYS (5.,1.,1.,1.,30.)
CALL UPEN1 (0.,0., 'DARROW')
CALL AXIS
CALL UEND
STOP
END
SUBROUTINE AXIS
CALL UMOVE (-1.,0.)
CALL UPEN1 (1.,0., 'LARROW')
CALL UMOVE (0.,-1.)
CALL UPEN1 (0.,1.)
RETURN
END

```

### **Subroutine UCOUNT**

#### **FUNCTION:**

This routine returns the length of the input data as the number of characters required when displaying the data using UPRINT.

#### **CALLING SEQUENCE:**

**CALL UCOUNT(DATA,COUNT)**

Where

**DATA** contains input information ready for display by UPRINT.

**COUNT** returns the number of characters which will be displayed when printing DATA on the display surface.

#### **OPTIONS which may apply:**

Alphameric Output Option: 'TEXT', 'REALNUMBER', 'INTERGERNUMBER',  
'XYCOORDINATES', 'XYZCOORDINATES'

Numeric Precision Option: UPSET option 'PRECISION'

#### **COMMENTS:**

More information about the option which apply may be found under UPRINT. In conjunction with UQUERY to determine current character size, this routine can be used to provide special character string positioning.

#### **Programming Notes:**



## **Subroutine UCRCL**

### **FUNCTION:**

This routine draws a circle whose center is at the point specified. The current beam position is not affected.

### **CALLING SEQUENCE:**

**CALL UCRCL (X,Y,RADIUS)**

Where

**X** is the X- or RADIUS coordinate in current user units

**Y** is the Y- or THETA coordinate in current user units.

**RADIUS** is the radius of the circle in current user units.

### **OPTIONS which may apply**

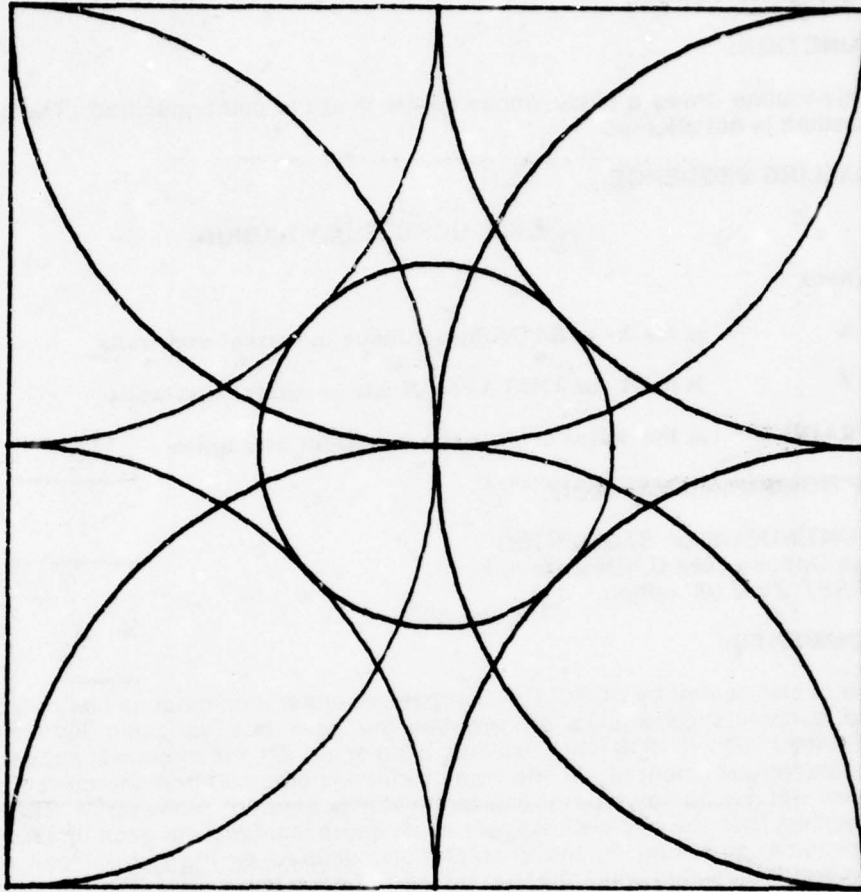
'CONTINUOUS' or 'SEGMENTED'  
Pen Options (See UPEN write up)  
UPSET 'ZVALUE' option

### **COMMENTS:**

The circle created by UCRCL will appear as either a continuous line or a collection of line segments depending on whether the user has specified 'CONTINUOUS' or 'SEGMENTED.' If 'CONTINUOUS' has been specified, the circle will appear as a single circular line segment of the line type of which is obtained from the current pen option. There will not be any type of line terminator generated. However, if 'SEGMENTED' is specified, then the circle will appear as an approximation with each line segment of the approximation taking on the characteristic defined by the current pen option. The terminator indicated will appear at the end of each line segment in the approximation.

*A call to UCRCL in a three dimensional graphics application program will result in a circle being drawn in the current X-Y plane, specified by the UPSET option 'ZVALUE'.*

### **Programming Notes:**



```

CALL USTART
CALL UOUTLN
CALL UCIRCLE (0.,0.,50.)
CALL UCIRCLE (50.,0.,50.)
CALL UCIRCLE (100.,0.,50.)
CALL UCIRCLE (100.,50.,50.)
CALL UCIRCLE (100.,100.,50.)
CALL UCIRCLE (50.,100.,50.)
CALL UCIRCLE (0.,100.,50.)
CALL UCIRCLE (0.,50.,50.)
CALL UCIRCLE (50.,50.,20.7)
CALL UEND
STOP
END

```

## **Subroutine UDAREA**

### **FUNCTION:**

This routine defines the boundaries of the device display area into which the output appearing in the user window into virtual space will appear.

### **CALLING SEQUENCE:**

**CALL UDAREA (XMIN,XMAX,YMIN,YMAX)**

Where

<b>XMIN</b>	is the minimum X-value in device absolute rectangular units in the SYSTEM coordinate system.
<b>XMAX</b>	is the maximum X-value in device absolute rectangular units in the SYSTEM coordinate system.
<b>YMIN</b>	is the minimum Y-value in device absolute rectangular units in the SYSTEM coordinate system.
<b>YMAX</b>	is the maximum Y-value in device absolute rectangular units in the SYSTEM coordinate system.

### **OPTIONS which may apply:**

'INCHES', 'CENTIMETERS', 'PERCENTAGEUNIT', 'RASTER', 'FONT'

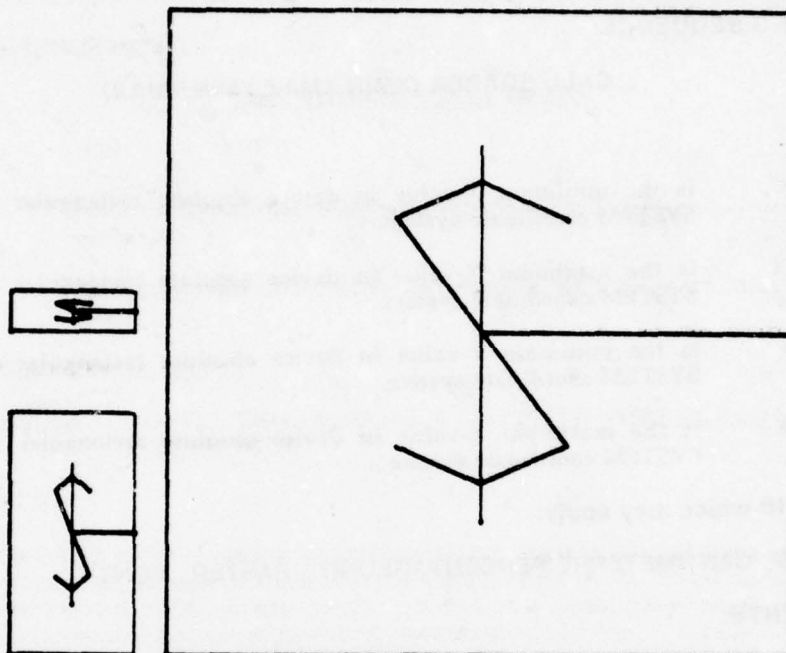
### **COMMENTS:**

This routine works in conjunction with UWINDO to define the mapping that takes place between virtual space and device space. UDAREA should be used for setting aside a specific region of the device display area for displaying graphical output created in the virtual space defined by UWINDO. While performing pen or beam movements in virtual space, any output which intersects the UWINDO boundaries will be truncated at that point. For a complete description of the windowing concept, see UWINDO.

The units used to describe the desired region of the display surface should be device, rectangular, and absolute. Any of the device units may be used (e.g., 'INCHES', 'CENTIMETERS,' etc.). If either the maximum X or maximum Y specification are less than the minimum X or minimum Y respectively, an error message will be generated (see Error Appendix for details) and the UDAREA specification in effect prior to entry to UDAREA will be retained.

### **Programming Notes:**





```

CALL USTART
CALL UDAREA (2.,12.,0.,8.)
CALL USET ('SOFTWARE')
CALL UPSET ('HORIZONTAL SIZE',70.)
CALL UPSET ('VERTICAL SIZE',90.)
CALL USET ('L&')
CALL UOUTLN
CALL UMOVE (100.,50.)
CALL UPEN (50.,50.)
CALL UDAREA (0.,1.0,0.,3.)
CALL U3WHER (X,Y,Z)
CALL UOUTLN
CALL UMOVE (100.,50.)
CALL UPEN (50.,50.)
CALL UDAREA (0.,1.0,4.,4.5)
CALL UOUTLN
CALL UMOVE (100.,50.)
CALL UPEN (50.,50.)
CALL UEND
STOP
END

```

**FUNCTION:**

This routine deletes an existing retained segment and removes its image from display surface.

**CALLING SEQUENCE:**

**CALL UDELET(SEGID)**

Where

**SEGID** is the segment identifier of the segment to be deleted (see UOPEN for a description of SEGID).

**OPTIONS which may apply:**

Segment/Frame Identifier Mode: 'FNAME', 'FNUMBER'  
Segment Posting Mode: 'DELAYED', 'IMMEDIATE'

**COMMENTS:**

The image of the deleted segment will be immediately removed, if visible, from the display surface if the segment posting mode is 'IMMEDIATE'. Otherwise, the image will not be removed until the next required screen erase. This will occur when UPOST or UERASE is executed or if UDELET or UMODFY (to change a segment attribute to 'INVISIBLE') is called with the segment posting mode as 'IMMEDIATE'.

**Programming Notes:**

## **Subroutine UDIMEN**

### **FUNCTION:**

This routine adjusts the physical boundaries of the display surface for those devices which have variable size plotting surface (e.g., drum plotters, or dot-matrix printers). All calls to this routine are ignored for devices having a fixed display surface.

### **CALLING SEQUENCE:**

**CALL UDIMEN (XMAX,YMAX)**

Where

**XMAX** is the maximum addressable coordinate in the X-direction in current device units.

**YMAX** is the maximum addressable coordinate in the Y-direction in current device units.

### **OPTIONS which may apply:**

Device unit type: 'INCHES', 'CENTIMETERS', 'FONTUNITS', 'PERCENTUNITS',  
'RASTERUNITS'

### **COMMENTS:**

Most display devices have fixed dimensions in which graphics output can be placed. For these devices, a call to UDIMEN will have no effect. However, for some devices such as drum plotters or dot-matrix printer which can plot on a whole roll of paper, it is usually convenient to select a portion of the roll of paper to which to plot. The maximum physical boundaries of this portion of the roll may be set by this routine. Normally UDIMEN should only be called immediately following a call to UERASE or USTART. For devices which are effected by a call to UDIMEN, the default display surface is the largest square which can be addressed using the standard size display medium.

### **CAUTION:**

*A call to UDIMEN will change the size of 'PERCENTUNITS'.*

**CCASSETTE CONTROL:** Not necessary to call USET ('DEVICE')

'RESTART' — Start read  
'RSTOP' — Stop read  
'WSTART' — Start write  
'WSTOP' — Stop write

### **Programming Notes:**



## Subroutine UDOIT

Interactive

### FUNCTION:

This subroutine enables the user to do various page layouts and control a cassette recorder such as the Tektronix 4923.

### CALLING SEQUENCE:

#### CALL UDOIT (ACTION)

Where

**ACTION** is a Hollerith string that indicates some function to perform

### ACTIONS

Must have called USET ('DEVICE')

'ALARM'	— Sounds the audible alarm
'ALPHAMODE'	— Sets alphamode
'AUDIBLE'	— Sounds the audible alarm
'BACKSPACE'	— Move back one character space
'BEEP'	— Sounds the audible alarm
'BELL'	— Sounds the audible alarm
'BLIP'	— Sounds the audible alarm
'BOTTOM'	— Move to bottom margin
'BSPb'	— Same as backspace
'CARR'	— Do a carriage-return
'CR'	— Do a carriage-return
'CRLF'	— Do a carriage-return then a line-feed
'DOWN'	— Do a line-feed
'ERAS'	— Erase the screen
'HDOWN'	— Home to the bottom
'HOME'	— Home to the top
'LEFT'	— Move to the left margin
'LFbb'	— Do a line-feed
'LINEFEED'	— Do a line-feed
'NEWLINE'	— Do a carriage-return then a line-feed
'NEWPAGE'	— Erase the screen
'PAGE'	— Erase the screen
'RIGHT'	— Move to right margin
'SPbb'	— Move over one hardware character space
'SPACE'	— Move over one hardware character space
'TABb'	— Tab to next horizontal tab setting
'TABH'	— Tab to next horizontal tab setting
'TOPb'	— Move to top margin
'UPbb'	— Move to top margin
'TABV'	— Tab to next vertical tab setting
'TABX'	
'TBHX'	— Tab to designated setting (example: 'TBH8')
'TBVX'	(see UPSET)
WHERE X = 0-9	
'UP'	— Move up one vertical hardware character position

**CASSETTE CONTROL: Not necessary to call 'USET ('DEVICE')**

'RSTART'	- Start read
'RSTOP'	- Stop read
'WSTART'	- Start write
'WSTOP'	- Stop write

## **Subroutine UDRAW**

### **FUNCTION:**

This routine draws a solid line vector from the current position to that specified by the input arguments. All lines are drawn on the XY-plane specified by the current set Z-value.

### **CALLING SEQUENCE:**

**CALL UDRAW(X,Y)**

Where

**X,Y** are the coordinates of the end point of the line in currency units

### **OPTIONS:**

Pen Coordinate Options:

Type: 'RECTANGULAR', 'POLAR'

Log: 'XLOG', 'YLOG', 'ZLOG', 'XYLOG', 'XZLOG', 'YZLOG', 'XYZLOG',  
'LOGARITHMIC', 'NOLOG'

MODE: 'ABSOLUTE', 'RELATIVE'

SPACE: 'VIRTUAL', 'DEVICE'

DEVICE SPACE UNITS: 'INCHES', 'CENTIMETERS', 'RASTERUNITS', 'FONTUNITS',  
'PERCENTUNITS', 'SPECIFICATIONUNITS'

COORDINATE SYSTEM: 'SYSTEM', 'USER', 'REFERENCE', 'WORKING'

### **COMMENTS:**

This routine is equivalent to UPEN with line option 'LNULL'. The current setting of the line option has no effect on this routine. *The current Z-value will be used to draw the line in three-space.*

### **Programming Notes:**



**FUNCTION:**

This routine provides a flexible drafting operation which obtains the end of the next beam/pen movement and the type of movement via graphics input, and then performs the operation requested. Only one operation will be performed by a single call to UDRIN. If the operation requested is not a valid UDRIN operation, then the input values are simply passed back to the calling program.

**CALLING SEQUENCE:**

**CALL UDRIN(X,Y,CHAR)**

Where

- |             |   |
|-------------|---|
| <b>X</b>    | <b>is the X- or RADIUS coordinate of the graphics input cursors in user units entered by the user during the graphics input operation.</b>              |
| <b>Y</b>    | <b>is the Y- or THETA coordinate of the graphics input cursors in user units entered by the user during the graphics input operation.</b>               |
| <b>CHAR</b> | <b>is the character entered which identifies the type of operation performed. It will be returned in Hollerith (left justified, blank-fill) format.</b> |

**OPTIONS which may apply:**

Any option which effects pen/beam.

Input Device Selection Options: 'LIGHTPEN', 'CURSORS', etc. (see UGRIN)

**COMMENTS:**

UDRIN supports the most basic line options for interactive drafting. When UDRIN is called, the current graphics input device (see UGRIN) is enabled. The user positions the cursor or tracking cross at the position he desires and strikes the particular keyboard key which indicates the option he desires. The following table indicates the UDRIN options available:

- A — draws an arrow from current beam location to current cursor position.
- C — outputs the current system character at the current cursor location.
- D — draws a double arrow from current beam location to cursor position.
- E — erases the screen.
- H — sets Graphic Status Area to hardware character type.
- K — sets the system character to that character entered at next appearance of cursors.
- L — draws a line from current beam position to cursor position.
- M — moves the beam invisibly to the cursor position.
- P — plots a point at cursor position.
- S — sets Graphics Status Area to software character type.

T — draws a ticked line from current beam location to cursor position.

X — prints coordinates at cursor position.

Z — draws a dashed line according to the current dash specification from the current beam position to the cursor position.

All unused characters may be assigned special meanings by the user to support options not available in UDRIN. The characters indicated above are all upper case, thus all lower case characters are available to the user.

**Subroutine UEND****FUNCTION:**

The routine should be called to terminate graphics operations. Its effect is to flush any output buffers which may be used internally in GCS, take the terminal out of graphics mode, display any error messages whose output has been deferred, position the beam or pen at the home position, and return to the user's program.

**CALLING SEQUENCE:**

**CALL UEND**

**OPTIONS which may apply:**

No options apply.

**COMMENTS:**

Designed as the means of terminating graphics operations in a program, UEND is the antithesis of USTART. Depending on the type of graphics device being used, different operations will take place which flush any buffers which are partially filled (see UFLUSH), disable the graphics operations at the terminal, position the pen or beam in the home position, and dump any GCS error messages (see Error Appendix) which have been deferred. UEND then returns control to the user's program. The user, if he wishes to perform additional graphics operations, should call USTART in order to reenter graphics mode (see USTART).

**Programming Notes:**



## **Subroutine UERASE**

### **FUNCTION:**

This routine causes the graphics display device plotting area to be cleared of any previous output. The pen or beam position is not affected.

### **CALLING SEQUENCE:**

**CALL UERASE**

### **OPTIONS which may apply:**

Security Banners: 'UNSECURED', 'UNCLASSIFIED', 'CONFIDENTIAL', 'SECRET',  
'TOP SECRET'

### **COMMENTS:**

The currently specified display surface(s) will be cleared. The pen or beam position at the time of the erasure will not be changed and the graphics status area will not be affected.

If a security banner other than 'UNSECURED' is selected, the banner will be displayed at the top and bottom of each frame in the largest hardware character size.

### **Programming Notes:**

## **Subroutine UERROR**

### **FUNCTION:**

This routine returns the identifier of the last GCS error which occurred and a count of the total number of GCS errors which have occurred.

### **CALLING SEQUENCE:**

**CALL UERROR(ERLAST,TOTAL)**

Where

**ERLAST** is the GCS error number of the last GCS error which occurred. Zero is returned if no errors have occurred.

**TOTAL** is the number of GCS errors which have occurred since the last GSA initialization (USTART or URESET) or the last call to UERROR whichever occurred most recently.

### **OPTIONS which may apply:**

No options apply

### **COMMENTS:**

The error total is reset to zero whenever UERROR is called.

There are three modes of error processing in GCS: 'ERRORS', 'SUPPRESSED ERRORS', and 'DEFERRED ERRORS'. The default mode is 'ERRORS' which indicates that error messages are to be generated as they occur. 'SUPPRESSED ERRORS' will not be generated at all. The first 10 'DEFERRED ERRORS' will be generated when UEND is called. The GCS Error Mode does not effect the execution of UERROR. Regardless of error mode, errors will be counted and the last error to occur will be saved.

### **Programming Notes:**

**Subroutine UFLUSH****FUNCTION:**

This routine terminates the frame.

**CALLING SEQUENCE:**

**CALL UFLUSH**

**OPTIONS which may apply:**

No options apply.

**COMMENTS:**

This routine must be called to terminate each frame. If not then plot data may be lost.

**Programming Notes:**



## **Subroutine UFRAME**

### **FUNCTION:**

This routine indicates to GCS that all graphic commands generated subsequent to the invocation of this subroutine and prior to the invocation of the matching subroutine UFREND are a named entity which replaces any previous occurrence of the same named entity.

### **CALLING SEQUENCE:**

**CALL UFRAME (NAME)**

Where

**NAME** is the eight character alphanumeric constant or variable.

### **OPTIONS which may apply:**

'INVISIBLE'

'VISIBLE'

### **COMMENTS:**

The execution of this subroutine causes all subsequent graphic commands to be associated with the argument NAME. The association continues until the matching subroutine UFREND is executed. For refresh graphic terminals, the framed information replaces the previous occurrence of the frame on the display screen. If GCS is in INVISIBLE build mode (default) the previous occurrence is deleted only when the new occurrence is completely defined. That is, when the matching subroutine UFREND is executed. If GCS is in visible build mode, the previous frame is deleted when UFRAME is executed and the new frame is displayed as each graphic command is generated. Multiple frames can be defined, but only one frame may be open or active at a time. This means that frame definitions may not overlap or be nested.

### **Programming Notes:**

### **Subroutine UFREND**

#### **FUNCTION:**

This routine indicates to GCS that the named set of graphic commands currently being defined (the active frame) is terminated.

#### **CALLING SEQUENCE:**

**CALL UFREND (NAME)**

Where

**NAME** is an eight character alphanumeric constant or variable.

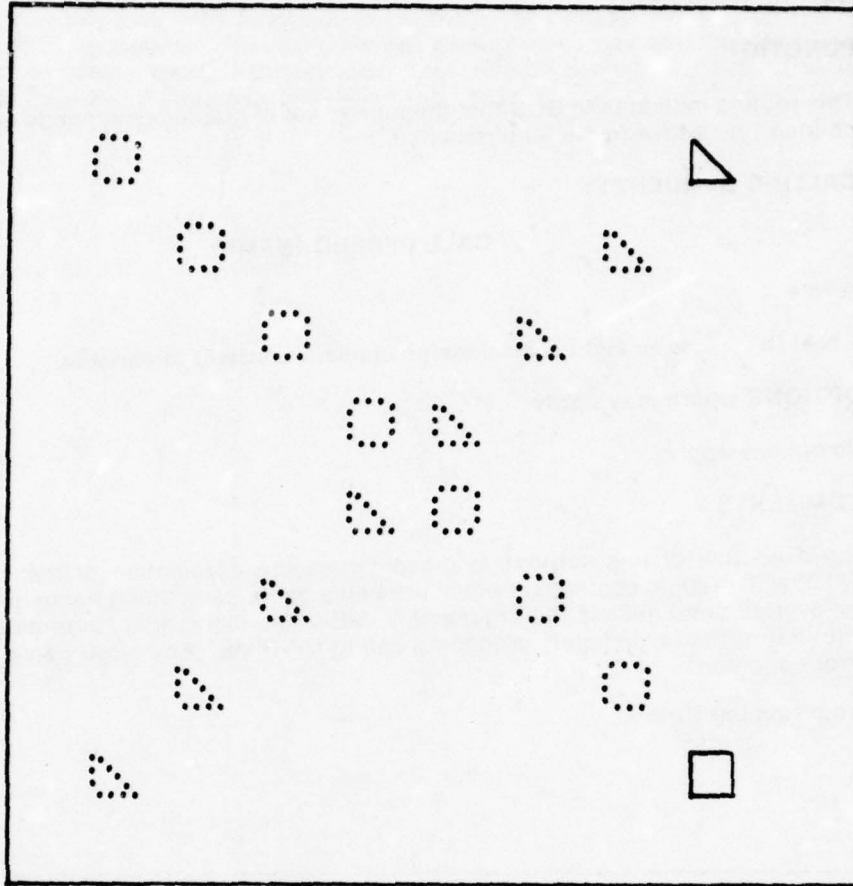
#### **OPTIONS which may apply:**

No options apply.

#### **COMMENTS:**

The execution of this subroutine discontinues the association of the current frame **NAME** with graphic commands which are being generated. Since frame definitions may not overlap or be nested, the argument **NAME** in the call to this subroutine must agree with that in the immediately preceding call to **UFRAME**. Any other name indicates an error condition.

#### **Programming Notes:**



```

X = 10.0
Y = 80.0
CALL USTART
CALL UPSET ('LIBRARY',1.)
CALL UOUTLN
DO 1 I = 1, 8
CALL UFRAME ('TRIANGLE')
CALL UMOVE (X,X)
CALL UPEN (X,(X+5.0))
CALL UPEN ((X+5.0),X)
CALL UPEN (X,X)
CALL UFREND ('TRIANGLE')
CALL UFRAME ('SQUARE')
CALL UMOVE (X,Y)
X = X + 10.0
Y = Y - 10.0
CALL URECT ((X-5.0),(Y+15.0))
CALL UFREND ('SQUARE')
1 CONTINUE
CALL UEND
STOP
END

```



**FUNCTION:**

This routine activates a graphics input device at the terminal and returns to the user the coordinates specified and an alphanumeric character which may be specified by the user or derived from the actions of the input device.

**CALLING SEQUENCE:**

**CALL UGRIN (X,Y,ICHAR)**

Where

<b>X</b>	is the X- or RADIUS coordinate of the point specified by the input device in user units.
<b>Y</b>	is the Y- or THETA coordinate of the point specified by the input device in user units.
<b>ICHAR</b>	is the alphanumeric entered by the user or provided by the device in Hollerith (left-justified, blank-filled) format.

**OPTIONS which may apply:**

'CURSOR', 'JOYSTICK', 'LIGHTPEN', 'TABLET', 'KEYBOARD', 'FUNCTIONKEY', 'MOUSE', 'BALL'.

Graphic Input Device Options

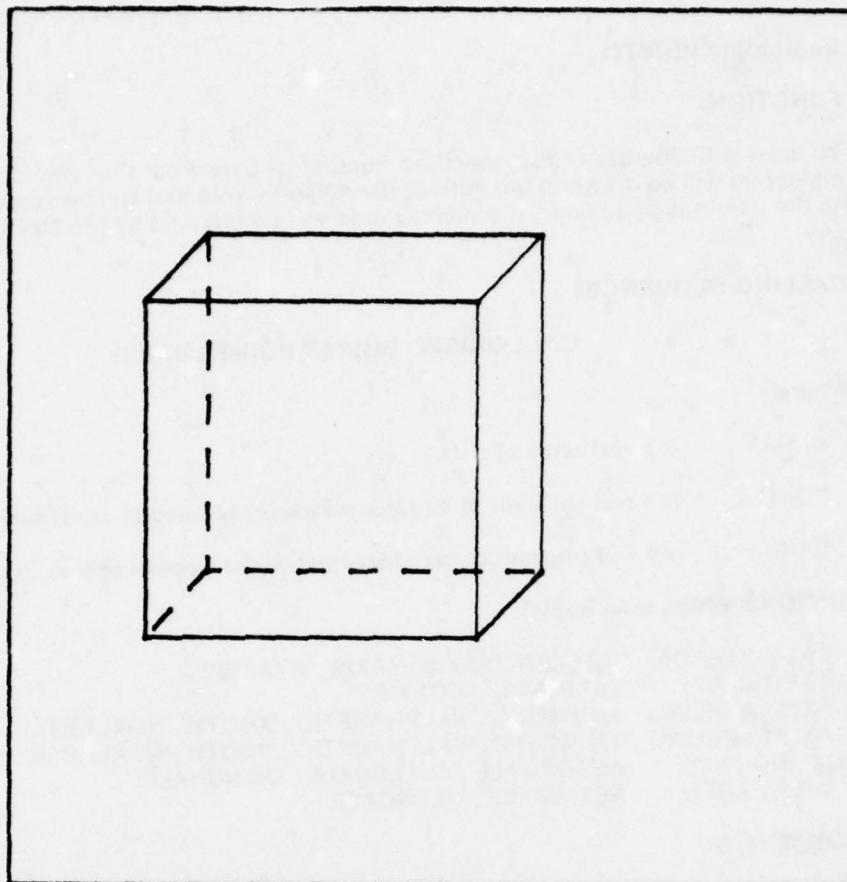
**COMMENTS:**

UGRIN is the user's general purpose graphics input routine. It is designed to support any of the input devices which may be available on any particular terminal. For terminals with more than one graphics input device, each type of device can be supported separately. Should the user request a device which is not available, the principal graphic input device will be enabled. Terminals with no graphics input device simulate graphics input by reading the coordinates and the character from the keyboard. In this case, the coordinates must be entered in *absolute, rectangular, raster units*.

The graphics input device which the user wishes enabled should be requested prior to calling UGRIN by setting the USET option for the device. A list of devices is indicated under 'Options which may apply' above. USTART will initialize the device selection to the primary graphics input device for the terminal.

When UGRIN is called, an indication that the graphics input device has been enabled will appear. This indication may be a tracking cross or cursors which appear on the screen, a ready light which turns on, or some other recognizable object. At this point, the user should position the cursor or tracking cross to the desired location and signal the end of the operation in the manner required by the device. This may consist of striking a keyboard character, removing the input device (pencil) from the input surface, applying pressure to the input device pencil, or some other means. For exact details for a particular device, see the System Manual. If graphics input is being simulated, enter the coordinates and the character from the keyboard upon request from the program. When the end of graphics input has been signaled, the graphics input enabled indicator will disappear.

The coordinates returned to the user's program in arguments X and Y will be in the units which are currently set in the Graphics Status Area. These values are suitable for direct entry as UPEN arguments. The character which has been entered will be in standard FORTRAN hollerith format (left-justified, blank-filled) and is suitable for use in FORTRAN LOGICAL IF statements for interrogation or as input to USET or UAOUT.



```
CHARACTER CHAR#1  
CALL USTART  
CALL UOUTLN  
1 CALL UGRIN (X,Y,CHAR)  
  IF (CHAR .EQ. 'S') CALL UPEN1 (X,Y,'LINE')  
  IF (CHAR .EQ. 'I') CALL UMOVE (X,Y)  
  IF (CHAR .EQ. 'D') CALL UPEN1 (X,Y,'DASH')  
  IF (CHAR .EQ. 'E') GO TO 2  
  GO TO 1  
2 CALL UEND  
  STOP  
  END
```

## **Subroutine UHISTO**

### **FUNCTION:**

To draw a histogram of the specified number of bars from the user data array. The histogram will be drawn in the current user display area and will be scaled and labeled as the user has specified. The display area will be reduced by the size of the labels, if any.

### **CALLING SEQUENCE:**

**CALL UHISTO (ARRAY,POINTS,BARS)**

Where

**ARRAY** is a real array of data.

**POINTS** is a real constant or variable which is the number of elements in **ARRAY**.

**BARS** is a real constant or variable which is the number of bars in the histogram.

### **OPTIONS which may apply:**

**AXIS EXISTENCE:** 'NOAXES', 'XAXIS', 'YAXIS', 'XYAXIS'

**AXIS FORMAT:** 'PLAINAXIS', 'TICAXIS'

**X-AXIS LABELING:** 'XNUMERIC', 'XALPHABETIC', 'XBOTH', 'NOXLABEL'

**Y-AXIS LABELING:** 'YNUMERIC', 'YALPHABETIC', 'YBOTH', 'NOXLABEL'

**SCALING TYPE:** 'AUTOSCALE', 'FULLSCALE', 'OWNSCALE'

**DISPLAY AREA:** 'NEWSCALE', 'OLDSCALE'

### **COMMENTS:**

UHISTO is a general purpose histogram drawing and labeling routine. The options indicated above are independent of each other so that several different options may appropriately be chosen. When the statistics for the histogram are developed, each interval is assumed to be closed on the lower end and open on the upper end. The last interval is closed on the upper end. That is, for example, the value of 2.0 will fall in the range of 2.0 to 3.0 rather than 1.0 to 2.0.

### **Programming Notes:**



## **SUMMARY OF OPTIONS:**

### **Axis Existence:**

- |           |  |
|-----------|--|
| NOAXIS    | — Neither the X-axis of the Y-axis nor the histogram will be displayed.  |
| XAXIS     | — Only the X-axis will be displayed.   |
| YAXIS     | — Only the Y-axis will be displayed.   |
| XYAXIS    | — Both the X- and the Y-axis will be displayed.  |
| PLAINAXIS | — The X-axis will appear as a solid line.  |
| TICAXIS   | — The X-axis will appear as a ticked line. The tic interval will be one unit. Note that only the X-axis format is involved. The Y-axis will always remain plain. The GRIDAXIS option is not supported. If GRIDAXIS is selected, the option will revert to PLAINAXIS. |

### **X-Axis Labeling:**

- |               |   |
|---------------|---|
| NOXLABEL      | — No labeling should appear along the X-axis  |
| XNUMERIC      | — Numeric labels should be created along the X-axis. The tic interval will be determined by UHISTO so that the number of labels which appear on the X-axis will be maximized.   |
| XALPHANUMERIC | — Alphanumeric labels will appear at the lower edge of the display area in which the histogram is drawn. The contents of the label can be specified by calling subroutine UPSET with arguments 'XLABEL' and a GCS character string as a value (for a description of GCS character strings, see UPRINT). |
| XBOTHLABELS   | — Both the alphanumeric and numeric labels will appear along the X-axis.  |

### **Y-Axis Labeling:**

- |             |  |
|-------------|--|
| NOYLABEL    | — No labeling should appear along the Y-axis.  |
| YNUMERIC    | — Numeric labels should be created along the Y-axis. One label will be created for each bar of the histogram. The numeric labels consist of the range of values represented by a histogram bar, and the count of the number of items in the range (the length of the bar).   |
| YALPHAMERIC | — Alphameric labels should be created along the Y-axis, at the left edge of the display area in which the histogram is drawn. The contents of the label can be specified by calling subroutine UPSET with arguments 'YLABEL' and a GCS character string as a value (for a description of GCS character strings, see UPRINT). |

**Scaling Type:****AUTOSCALE**

- For the X-axis, the range of values for the histogram will be from zero to a number larger than the size of the largest bar. For the Y-axis, the range of values to be histogrammed is examined so that 'nice' numbers will appear at the bar intervals. The range may be expanded to include these values. The UWINDO is set to the expanded range as identified.

**FULLSCALE**

- For the X-axis, the range of values for the histogram will be from zero to a number larger than the size of the largest bar. For the Y-axis, the limits of the histogram will be the minimum and maximum values of the data. The UWINDO setting is modified to reflect the limits of the X-axis.

**OWNSCALE**

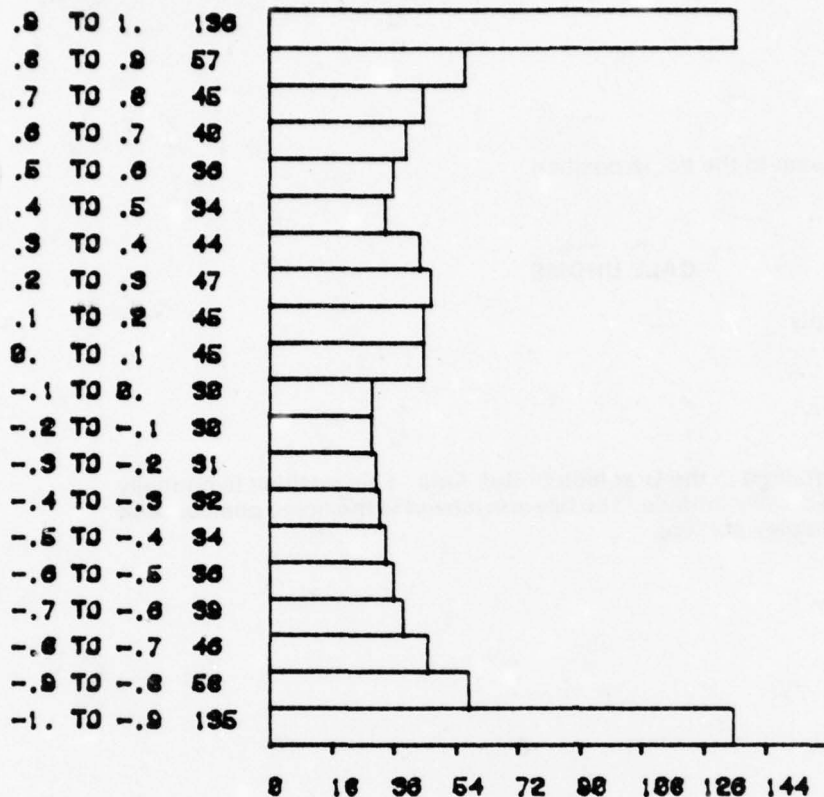
- For the X-axis, the left edge of the window setting is used to determine the lower limit of the histogram. The upper limit is set to be larger than the lower limit and larger than the size of the largest bar. The UWINDO setting is modified to reflect the limits of the X-axis. For the Y-axis, the limits of the histogram will be the upper and lower Y limits of the window. Histogram bars outside of the limit are windowed.

**Display Area:****NEWSCALE**

- The histogram is considered to be drawn in a 'new' area. The appropriate scaling is performed and the specified labels are produced.

**OLDSCALE**

- The histogram is considered to be drawn in an 'existing' display area in which another histogram or plot may exist. No labels are produced and no rescaling occurs. This option is an override option over the previous alternatives.



```

DIMENSION DATA(1000)
DATA XN/1000./
DO 2 I = 1, 1000
2 DATA(I) = SIN(FLOAT(I)/150.)
CALL USTART
CALL UASPCT (1.)
CALL USET ('LARGE')
CALL USET ('PERCENTUNITS')
CALL UDAREA (0.,100.,0.,100.)
CALL USET ('FULLSCALE')
CALL USET ('XBOTHLABELS')
CALL UPSET ('XLABEL','DISTRIBUTION OF VALUES OF SINE\')
CALL UHISTO (DATA,XN,20.)
CALL UEND
STOP
END

```

**Subroutine UHOME****FUNCTION:**

This routine moves the beam to the home position.

**CALLING SEQUENCE:**

**CALL UHOME**

**OPTIONS which may apply:**

No options apply.

**COMMENTS:**

The home position is maintained in the Graphics Status Area. This position is normally the lower left corner of the display surface. The beam is moved to the home position with no visible output on the display surface.

**Programming Notes:**



Subroutine UIMAGE

3D

**FUNCTION:**

This routine applies a general two-dimensional image transformation to the indicated segment/frame.

**CALLING SEQUENCE:**

**CALL UIMAGE(X,Y,SX,SY,R,SEGID)**

Where

**X,Y** is the new position of the segment in current 2D device units  
**SX,SY** is the scale factor to be applied along each axis of the display surface  
**R** is the rotation in current angular units to be applied around the Z-axis of the display surface  
**SEGID** is the identifier of a 'retained' segment/frame which was UOPENed for at least general 2D image transformations

**OPTIONS which may apply:**

Device Units: 'INCHES', 'CENTIMETERS', 'RASTER UNITS', 'FONT UNITS', 'SPECIFICATION UNITS', 'PERCENT UNITS'

Specification Units (UPSET): 'SPECIFICATION UNITS', 'XSPECIFICATION UNITS', 'YSPECIFICATION UNITS', 'ZSPECIFICATION UNITS'

Angular Units: 'DEGREES', 'RADIANS', 'PIRADIANS', 'GRADS', 'MILS'

Segment Identifier Mode: 'FNAME', 'FNUMBER', 'SNAME', 'SNUMBER'

**COMMENTS:**

The image transformation is applied in the order: rotation scaling, and translation. If the resulting image should exceed the display surface address space, the result is undetermined.

Image transformations are only applied if supported on the current display surface. Requests for image transformations will be ignored if the display device does not support this facility. 2D transformations will be applied if 3D transformations are supported.

**Programming Notes:**

AD-A063 167

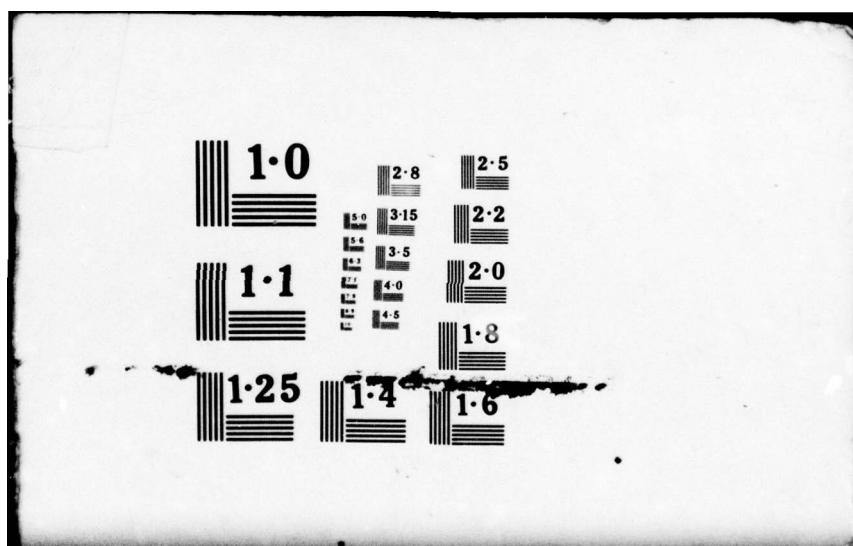
ARMY ENGINEER WATERWAYS EXPERIMENT STATION VICKSBURG MISS. F/G 9/2  
GRAPHICS COMPATIBILITY SYSTEM (GCS). PRIMER ON COMPUTER GRAPHIC--ETC(U)  
1978

UNCLASSIFIED

NL

4 OF 5  
AD A  
063167





**FUNCTION:**

This routine enables the user to read character data entered from the alphanumeric key board of the terminal. The data will be formatted and returned to the user in one of five options. The options are: 'TEXT', 'REALNUMBER', 'INTEGER', OR 'XYCOORDINATE', 'XYZ COORDINATE'.

**CALLING SEQUENCE:**

**CALL UINPUT (DATA,COUNT,FLAG,OPTION)**

Where

- DATA** is a single variable or array to contain either real number or a GCS text string. The size of DATA is variable: it is a single variable if 'REAL' or 'INTEGER' is specified; it is a two word array if 'XYCOORDINATES' is specified; and it is of variable length if 'TEXT' is specified.
- COUNT** is a single variable or constant. If the user has specified 'TEXT' input, then COUNT is the maximum number of characters returned in DATA. The size of DATA must be large enough to contain COUNT characters. The characters are returned in Hollerith format; left justified and blank filled. If the terminal operator inputs fewer characters than requested in COUNT, then DATA will be blank filled to the required size. If the user has specified 'REAL', 'INTEGER', 'XYCOORDINATE' or 'XYZCOORDINATE' then the value of COUNT upon entry to UINPUT is the number of variables to be input. The size of DATA must be large enough to contain COUNT variables.
- FLAG** is a single variable. The value of FLAG, upon exit from UINPUT, will be negative if the terminal operator has made an error in entering the numeric data. In this case, the value of DATA is undefined. If the required input was correct, then the value of FLAG is the number of elements returned in Data. The value of FLAG is always less than or equal to COUNT.
- OPTION** is a character constant or variable which specifies the format into which the input string will be mapped. The format options are: 'TEXT', 'REAL', 'INTEGER', 'XYCOORDINATE' or 'XYZCOORDINATE'.

**OPTIONS which may apply:**

Device Switching Options: 'MESSAGEDEVICE', 'PLOTDEVICE'

**COMMENTS:**

The input begins at the current beam or alphanumeric cursor position. Upon termination of UINPUT, the beam position in effect prior to its invocation is restored. For a full explanation of the GCS alphanumeric input facility, see SUBROUTINE UREAD.



ENTER: HELLO

ENTER: 1.234E+10

ENTER: -123456789

ENTER: 1.2,3.4

HELLO .1234E+11 -123456789 (1.2,3.4)

```
CHARACTER OPTION*4(4)
DIMENSION COUNT(4),DATA(6),INDEX(4)
DATA COUNT,INDEX,X,Y/5.,1.,1.,1.,1.,3.,4.,5.,5.,99./
DATA OPTION/'TEXT','REAL','INTE','XYCO'/
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UOULN
DO 1 I = 1, 4
  IDX = INDEX(I)
  CALL UMOVE (X,Y)
  CALL UPRNT1 ('ENTER: ',',','TEXT')
  CALL UINPUT (DATA(IDX),COUNT(I),FLAG,OPTION(I))
  IF(I.EQ.1)CALL UAPEND(COUNT(1),DATA(INDEX(1)),DATA(INDEX(1)))
  CALL USET (OPTION(I))
  CALL UPRINT (X,10.,DATA(IDX))
  X = X + 22.5
  Y = Y - 28.
1 CONTINUE
CALL UEND
STOP
END
```

### **Subroutine UINVOK**

#### **FUNCTION:**

This invokes an already constructed GCS data structure. The current beam/pen position becomes the origin of the structure's coordinate system.

#### **CALLING SEQUENCE:**

**CALL UINVOK (NAME)**

Where

**NAME** is an eight-character GCS structure name of an already defined structure.

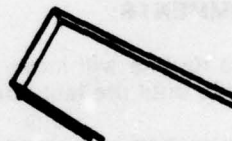
#### **OPTIONS which may apply:**

Invocation Scaling Options (UPSET): 'XSCALE', 'YSCALE', 'ZSCALE'

Invocation Rotation Options (UPSET): 'XROTATION', 'YROTATION', 'ZROTATION'

#### **COMMENTS:**

The scaling and rotation values for each axis of the structure's coordinate system are obtained from the current settings of the Invocation Scaling and Rotation Options. The origin of the structure's coordinate system is set to the current beam/pen position. A call to UINVOK is equivalent to a call using relative coordinates to U3CALL (0,0,-0,SX,SY,SZ,RX,RY,RZ,NAME).



```

CALL ATTACH (2, '/TEKSDENO/SAVE', '3,8,IST,')
CALL USTART
CALL UASPCT (1.)
CALL UUNDO (-100., 100., -100., 100.)
CALL UPSET ('LIBRARY FILE', 1.)
CALL UTILITY ('LOAD', 2.)
CALL USMOVE (-50., 5., 0.)
CALL UINVOK ('BOX ')
CALL UPSET ('XSCALE', 2.)
CALL USMOVE (5., 5., 0.)
CALL UINVOK ('BOX ')
CALL UPSET ('ZROTATION', 45.)
CALL USMOVE (-50., -50., 0.)
CALL UINVOK ('BOX ')
CALL UPSET ('XROTATION', 25.)
CALL UPSET ('YROTATION', 25.)
CALL UPSET ('YSCALE', .5)
CALL USMOVE (50., -50., 0.)
CALL UINVOK ('BOX ')
CALL UEND
STOP
END

```



## **Subroutine ULINE**

### **FUNCTION:**

This routine creates a line string on the current XY-plane by connecting an array of coordinates with lines drawn with the current line option.

### **CALLING SEQUENCE:**

**CALL ULINE (X,Y,PTS)**

Where

**X** is an array of length PTS containing the X-components of the coordinates of the points to be connected.

**Y** is an array of length PTS containing the Y-components of the coordinates of the points to be connected.

**PTS** is a real variable which specifies the number of points to be connected.

### **OPTIONS which may apply:**

All pen-related options (see UPEN)

### **COMMENTS:**

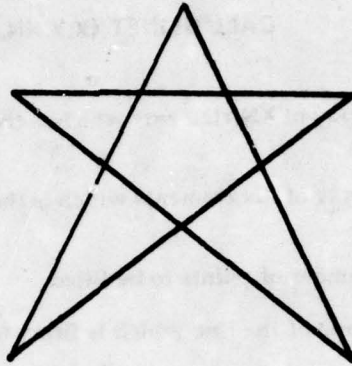
This routine will move to the first point specified and then draw a line to succeeding points until the last point is reached. The pen or beam will be left at the last point.

This routine is the complement of the UAXIS routine. UAXIS may be used to create an axis and a plotting environment. ULINE can then be used to plot within this environment.

*In three-dimension applications, the line drawn by ULINE will lie in the current XY plane.*

### **Programming Notes:**





```
DIMENSION X(6),Y(6)
DATA X/30.,50.,70.,30.,70.,30./
DATA Y/30.,70.,30.,60.,60.,30./
CALL USTART
CALL ULINE (X,Y,6.)
CALL UEND
STOP
END
```

### **Subroutine ULINFT**

#### **FUNCTION:**

This routine fits a linear equation to the specified input data in the least squares sense and returns the slope and intercept of the equation.

#### **CALLING SEQUENCE:**

**CALL ULINFT (X,Y,XN,S,YI)**

#### **Where**

- X** are an array of XN elements which is the X component of the data points to be fitted.
- Y** are an array of XN elements which is the Y component of the data points to be fitted.
- XN** is the number of points to be fitted
- S** is the slope of the line which is fitted to the data points.
- YI** is the intercept of the line which is fitted to the data points.

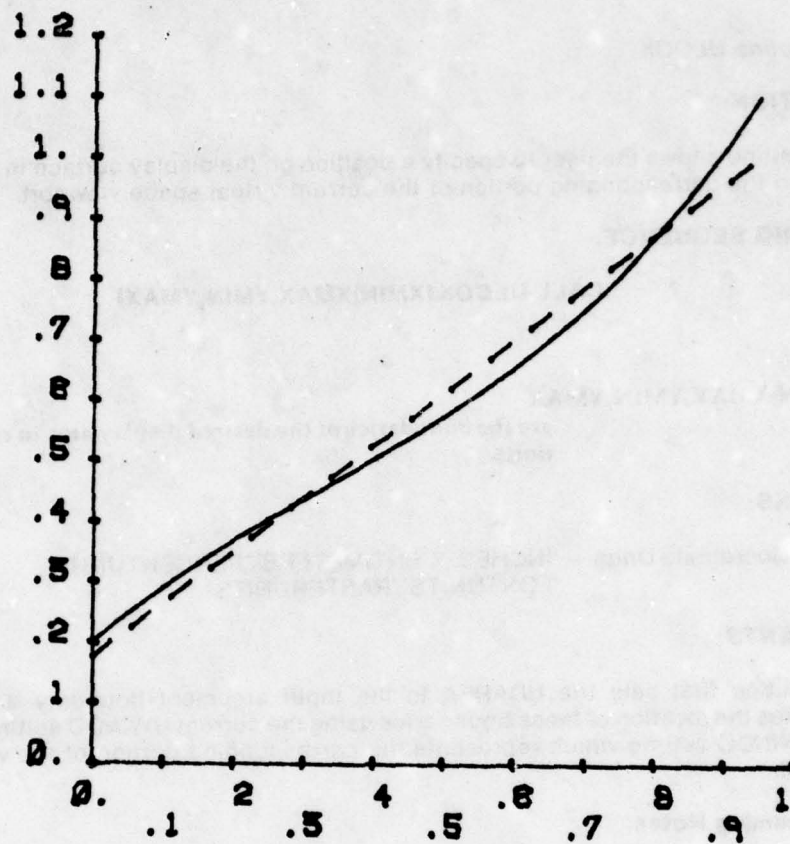
#### **OPTIONS which may apply:**

No options apply.

#### **COMMENTS:**

The number of points to be fitted must be greater than one.

#### **Programming Notes:**



```

DIMENSION X(20),Y(20)
DO 10 I = 1, 20
  X(I) = FLOAT(I-1) / 20.
10 Y(I) = X(I)**3 - X(I)**2 + X(I) + .2
CALL USTART
CALL USET ('PERCENT UNITS')
CALL USET ('SOFTWARE CHARACTERS')
CALL UPSET ('HORIZONTAL',.83)
CALL UPSET ('VERTICAL',.85)
CALL UPLOT (X,Y,1..20., 'LINE')
CALL ULINFT (X,Y,20.,SLOPE,YI)
CALL UNOVE (0.,YI)
XI = 10. / 20.
CALL USET ('DASHED LINES')
CALL UPEN (XI,SLOPE*XI+YI)
CALL UEND
STOP
END

```



**FUNCTION:**

This routine allows the user to specify a position on the display surface in which will be mapped the corresponding portion of the current virtual space viewport.

**CALLING SEQUENCE:**

**CALL ULOOK(XMIN,XMAX,YMIN,YMAX)**

Where

**XMIN,XMAX,YMIN,YMAX**

are the boundaries of the desired display area in current device units.

**OPTIONS:**

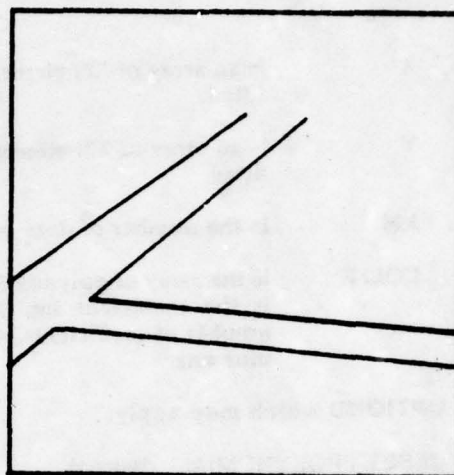
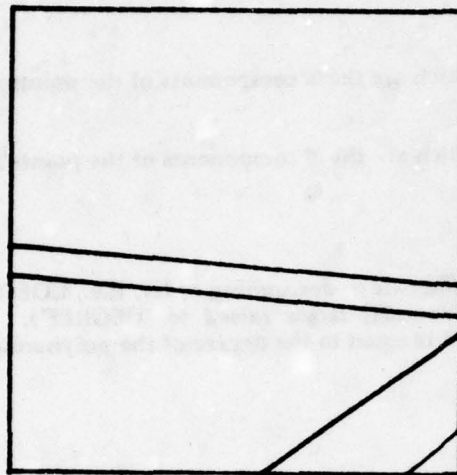
Device Coordinate Units — 'INCHES','CENTIMETERS','PERCENTUNITS',  
'FONTUNITS','RASTERUNITS'

**COMMENTS:**

This routine first sets the UDAREA to the input argument boundary area. It then calculates the location of these boundaries using the current UWINDO setting to obtain a new UWINDO setting which represents the corresponding portion of the virtual space viewport.

**Programming Notes:**





```

CALL ATTACH (2, '/TEKSDemo/SAVE', 3, 0, 1ST, )
CALL USTART
CALL UPSET ('LIBRARY', 1.)
CALL UTILITY ('LOAD', 2.)
CALL UWINDO (-100., 100., -100., 100.)
CALL UVIEW (100., -250., 0., 0., 100., 0.)
CALL ULOOK (3.5, 0.5, 1., 4.)
CALL UOUTLN
CALL USCALL (0., 0., 0., 1., 1., 1., 0., 0., 0., 'ROAD ')
CALL ULOOK (0., 3., 1., 4.)
CALL UOUTLN
CALL USCALL (0., 0., 0., 1., 1., 1., 0., 0., 0., 'ROAD ')
CALL UEND
STOP
END

```

### **Subroutine ULSTSQ**

#### **FUNCTION:**

This routine fits a polynomial equation to the input data by the method of least squares and returns the coefficients of the fitted equation.

#### **CALLING SEQUENCE:**

**CALL ULSTSQ (X,Y,XN,COEFF)**

Where

<b>X</b>	is an array of XN elements which are the X components of the points to be fitted.
<b>Y</b>	is an array of XN elements which are the Y components of the points to be fitted.
<b>XN</b>	is the number of data points.
<b>COEFF</b>	is the array of polynomial coefficients in descending order, (i.e., COEFF(1)) is the coefficient for the polynomial factor raised to 'DEGREE'. The number of coefficients returned is equal to the degree of the polynomial fit plus one.

#### **OPTIONS which may apply:**

UPSET ('POLYNOMIAL', degree)

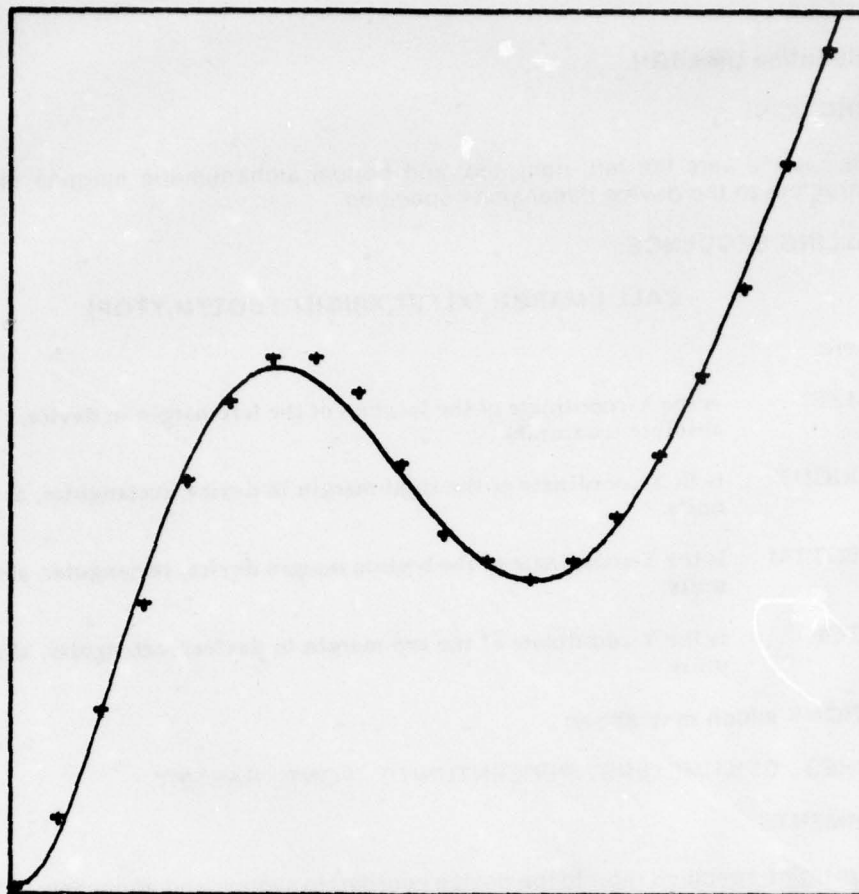
Where

degree is the degree of the desired polynomial fit.

#### **COMMENTS:**

A request for a polynomial fit of degree less than or one degree, or greater than ten is an error. The points to be fitted must be in ascending order. Least squares computations within the subroutine are carried out in double precision arithmetic. Users are cautioned against attempting to perform a high degree fit with a small number of data points.

#### **Programming Notes:**



```

PARAMETER IDEGRE=7
DIMENSION A(IDEGRE),X(20),Y(20)
DATA X/0.,5.,10.,15.,20.,25.,30.,35.,40.,45.,50.,55.,60.,
&      65.,70.,75.,80.,85.,90.,95./
DATA Y/0.,8.,22.,32.,46.,55.,60.,60.,58.,48.,37.,28.,
&      20.,15.,10.,10.,15.,20.,28.,37.,48./
CALL USTART
CALL UOULN
CALL UPSET ('POLYNOMIAL',FLOAT(IDEGRE-1))
DO 2 I =1, 20
  XN = FLOAT(I)
  CALL USET ('N+')
  CALL UPEN (XC(I),Y(I))
  CALL USET ('LINE')
2 CONTINUE
CALL UMOVE (0,0,0,0)
CALL ULSTSQ (X,Y,XN,A)
DO 5 I =1, 100
  Y0 = A(I)
  X0 = FLOAT(I)
  XK = X0
  DO 4 J = 2, IDEGRE
    Y0 = A(J) * XK + Y0
    XK = XK * X0
  4 CONTINUE
  CALL UPEN (X0,Y0)
5 CONTINUE
CALL UEND
STOP
END

```



## **Subroutine UMARGN**

### **FUNCTION:**

This routine sets the left, right, top, and bottom alphanumeric margins for hardware characters to the device dimensions specified.

### **CALLING SEQUENCE:**

**CALL UMARGN (XLEFT,XRIGHT,YBOTTM,YTOP)**

Where

<b>XLEFT</b>	is the X-coordinate of the location of the left margin in device, rectangular, absolute user units.
<b>XRIGHT</b>	is th X-coordinate of the right margin in device, rectangular, absolute user units.
<b>YBOTTM</b>	is the Y-coordinate of the bottom margin device, rectangular, absolute user units
<b>YTOP</b>	is the Y-coordinate of the top margin in device, rectangular, absolute user units.

### **OPTIONS which may apply:**

'INCHES', 'CENTIMETERS', 'PERCENTUNITS', 'FONT', 'RASTER'

### **COMMENTS:**

The margins specified refer to the device coordinate system and apply only to the output of hardware characters. Software characters will be affected by the margin settings when UPRINT/UPRNT is called using DEVICE coordinates. If the right margin position specified is less than the left margin position, the margin is not changed and an error message is generated (see Error Appendix). Similarly if the top margin position specified is not greater than the bottom margin, the margin is not changed and an error message is generated.

### **Programming Notes:**



THIS IS AN EXA  
MPLE OF THE MA  
RGINING EFFECT  
WITHIN A DEFINED UDAREA

```
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('PERCENT UNITS')
CALL UOUTLN
CALL UDAREA (50.,70.,40.,70.)
CALL UOUTLN
CALL USET ('DEVICE')
CALL UNARON (50.,70.,40.,70.)
CALL UPRINT (50.,70., 'THIS IS AN EXAMPLE OF THE MARGINING
&EFFECTS WITHIN A DEFINED UDAREA,')
CALL UEND
STOP
END
```

## Subroutine UMENU

*Interactive*

### FUNCTION:

This routine displays a menuboard containing up to ten options which are offered for selection. The user selects an option using the terminal's graphics input device (if it has one) or keyboard (if it doesn't). The number of the option selected is returned to the user.

### CALLING SEQUENCE:

**CALL UMENU (OPTNO,OPTNAM,SELECT)**

Where

- OPTNO** is a FORTRAN REAL variable whose absolute value indicates the number of options available and whose sign determines whether the menuboard already is being displayed. A positive value indicates that the menuboard should be created; a negative value indicates that the menuboard is already being displayed.
- OPTNAM** is an array containing the eight-character names of the options for this menu. These names need not have GCS string terminators. There must be one name for each option.
- SELECT** is the identifying number of the option chosen by the user which is returned to the calling program.

### OPTIONS which may apply:

Hardware Character Size: 'SMALL', 'MEDIUM', 'LARGE', 'EXTRALARGE' (see UPRINT).

Graphics Input Device: 'CURSORS', 'KEYBOARD', etc. (see UGRIN).

### COMMENTS:

The menuboard is normally displayed on the left side of the plotting surface. If the device has no graphics input device, then the menu is displayed alphanumerically on the control device. The identification numbers have been designed so that the value returned to the user is a REAL number whose INTEGER value is suitable for use as a COMPUTED GO TO index. If the user selects outside of the displayed menu elements, the user is asked to select an option again.

18  
 UNITS  
 9  
 PERCENT  
 8  
 LARGE  
 7  
 OPTION7  
 6  
 OPTIONS6  
 5  
 OPTIONS5  
 4  
 OPTION4  
 3  
 OPTIONS3  
 2  
 OPTION2  
 1  
 OPTION1

OPTION 6 SELECTED

```

DIMENSION MENU(18)
DATA MENU/'OPTION1','OPTION2','OPTIONS','OPTION4','OPTIONS',
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('LARGE')
CALL USET ('PERCENT UNITS')
CALL UDAREA (28.,99.,8.,199.)
CALL UOUTLN
CALL UMENU (18.,MENU,SELECT)
CALL UPRINT (25.,58.,'OPTION ',')
CALL UPRNT1 (SELECT,'INTEGER')
CALL UPRNT1 (' SELECTED',',','TEXT')
CALL UEND
STOP
END
  
```



**FUNCTION:**

This routine modifies the setting of segment attributes for existing retained segments.

**CALLING SEQUENCE:**

**CALL UMODFY(SEGID,ATID,ATVAL)**

Where

**SEGID** is the segment identifier of the segment whose attribute is to be changed (see UOPEN for a description of SEGID).

**ATID** is the attribute identifier of the segment attribute whose setting is to be modified. Only three segment attribute identifiers are allowed:

'VISIBILITY'  
'HIGHLIGHTING'  
'DETECTABILITY'

Only the first four characters are significant.

**ATVAL** is the value to which the indicated segment attribute is to be set. Valid values are 'ON' and 'OFF'. Any other value will generate an error and retain the existing setting.

**OPTIONS which may apply:**

Segment/Frame Identifier Mode: 'FNAME', 'FNUMBER'  
Segment Posting Mode: 'IMMEDIATE', 'DELAYED'

**COMMENTS:**

If the new segment attribute value involves removing information from the display surface, then this will occur immediately if in 'IMMEDIATE' segment posting mode. If in 'DELAYED' segment posting mode, the information will be removed at the next UPOST or UREASE, or UDELET or UMODFY in 'IMMEDIATE' segment posting mode, whichever occurs first.

Visibility specifies whether the segment is to be displayed or merely defined. Highlights will occur by blinking if possible on the selected display device. Otherwise, heightened intensity or wider lines may be substituted. Detectability controls whether the segment can be recognized during 'pick'-type input operations.

**Programming Notes:**



### **Subroutine UMOVE**

#### **FUNCTION:**

The beam/pen is positioned at the location specified in the arguments. No visible output will appear on the terminal.

#### **CALLING SEQUENCE:**

**CALL UMOVE (X,Y)**

Where

**X** is the X- or RADIUS coordinate in current user units.

**Y** is the Y- or THETA coordinate in current user units.

#### **OPTIONS which may apply:**

Pen Coordinate Options (see UPEN)

Coordinate System Options (see UCOSYS)

#### **COMMENTS:**

The execution of this subroutine has no effect on the pen status option (i.e., the line/terminator specification). The beam movement, although invisible is subject to windowing. The operation of UMOVE is identical to that of UPEN when the pen status specified in 'NNUL' (see UPEN).

#### **Programming Notes:**

## **Subroutine UNSAVE**

### **FUNCTION:**

This routine will restore the Graphics Status Area (GSA) to its condition at the time the array from which the status is obtained was loaded.

### **CALLING SEQUENCE:**

**CALL UNSAVE (SARRAY)**

Where

**SARRAY** is an array in which a previous state of the GSA was saved by **USAVE**.

### **OPTIONS which may apply**

No options apply.

### **COMMENTS:**

The user should avoid modifying the contents of the save array before he calls **UNSAVE** as this may cause indeterminate effects during subsequent graphics operations. The current pen position is restored to the position which is specified in the saved array.

### **Programming Notes:**

**Subroutine UNSHOW**

**FUNCTION:**

To request that the named frame be placed in omit status.

**CALLING SEQUENCE:**

**CALL UNSHOW (NAME)**

Where

**NAME** is an eight character alphanumeric constant or variable of a currently defined frame.

**OPTIONS which may apply:**

No options apply.

**COMMENTS:**

The execution of this subroutine causes the frame NAME to be deactivated on the face of the display screen. The frame must have been previously defined by a subroutine UFRAME and subroutine UFRAME and subroutine UFREND pair.

**Programming Notes:**

### **Subroutine UNSVPN**

#### **FUNCTION:**

This routine restores the state of all pen-related variables in the Graphics Status Area (GSA) to their condition at the time the array from which the status is obtained was loaded.

#### **CALLING SEQUENCE:**

**CALL UNSVPN (SARRAY)**

Where:

**SARRAY** is an array in which the state of all pen-related variables of the GSA was saved by USVPN.

#### **OPTIONS which may apply:**

No options apply.

#### **COMMENTS:**

For a description of when UNSVPN should be used, see USVPN. The user should avoid altering the contents of the save array before he calls UNSVPN as this may cause indeterminate effects during subsequent graphics operations. The current pen position is restored to the position which is specified in the saved array.

#### **Programming Notes:**



## **Subroutine UNSVTR**

### **FUNCTION:**

This routine will restore the user coordinate system transformation status of the Graphics Status Area (GSA) to its condition at the time the array from which the status is obtained was loaded.

### **CALLING SEQUENCE:**

**CALL UNSVTR (SARRAY)**

Where

**SARRAY** is an array in which a previous state of the coordinate system transform was saved by USVTR.

### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

The invocation of this subroutine will cause the automatic reactivation of the user coordinate system which was in effect at the time that the array was loaded. The user should avoid modifying the contents of the save area before he calls UNSVTR as this may cause indeterminate effects during subsequent graphics operations.

### **Programming Notes:**

## Subroutine UOPEN

3D

### FUNCTION:

This routine creates a segment/frame with the specified name and characteristics. Subsequent primitives will be inserted into the segment.

### CALLING SEQUENCE.

#### CALL UOPEN(SEGID)

Where

**SEGID** is a real number whose integer value is a positive integer (if in 'FNUMBER' mode) or is a Hollerith string of eight characters (if in 'FNAME' mode) which will be used as the segment identifier.

### OPTIONS which may apply:

Retained Segment Type:	'NOTTRANSFORMATION'	Segment cannot be transformed
	'2DTRANSLATION'	Segment can be translated in 2D
	'2DGENERAL'	Segment can be translated, scaled and rotated in 2D
	'3DTRANSLATION'	Segment can be translated in 3D
	'3DGENERAL'	Segment can be translated, scaled and rotated in 3D

Segment/Frame Identifier Mode:	'FNAME', 'FNUMBER'
Segment Retention Mode:	'RETAINED', 'NONRETAINED', 'STORAGE', 'REFRESHED'
Segment Visibility:	'VISIBLE', 'INVISIBLE'
Segment Highlighting:	'NOHIGHLIGHTING', 'HIGHLIGHTING'
Segment Detectability:	'UNDETECTABLE', 'DETECTABLE', 'SENSITIZED', 'DESENSITIZED'

### COMMENTS:

Segments may be created in either of two modes, 'RETAINED' ('REFRESHED') and 'NONRETAINED' ('STORAGE'). Segments which are 'RETAINED' may be transformed (if specified) and their segment attributes may be modified (see description of segment attributes below). Segments which are 'NONRETAINED' are displayed but their descriptions are not saved. Such segments will disappear when the display surface is erased or posted (see UPOST). The following discussion pertains only to 'RETAINED' segments.

Each 'RETAINED' segment has three segment attributes associated with it: visibility, highlighting, and detectability. The visibility attribute determines whether a segment is displayed or not. The highlighting attribute indicates whether the segment should be displayed in some enhanced mode. The detectability attribute governs the sensitivity of the segment to 'pick' input. When a segment is UOPENed, the segment attributes are set to current settings contained in the GSA. Segment attributes may be modified at any time by the UMODIFY (q.v.) function.

After the segment has been opened, the current value of the following GCS modes will be placed in the segment:

line width  
intensity/brightness

line style  
text angle  
hardware character size  
text font

Image transformations may be applied to 'RETAINED' segments if the display device is capable of supporting these transformations. Attempts to perform image transformations on devices which do not have the capability will be ignored and the segment will remain untransformed. It should be noted that image transformations involve moving images around on the display surface and are not the same as moving objects in virtual space. Four routines are provided for applying image transformations as indicated:

Transformation	Applied By
'2DTRANSLATION'	UPLACE(X,Y,NAME)
'2DGENERAL'	UIMAGE(X,Y,SX,SY,R,NAME)
'3DTRANSLATION'	U3PLAC(X,Y,Z,NAME)
'3DGENERAL'	U3IMAG(X,Y,Z,SX,SY,SZ,RX,RY,RZ,NAME)

Image transformation may only be applied to a retained segment which has been 'UOPEN'ed with the retained segment type mode set to a type at least as general as the image transformations to be performed on the segment. The default type is 'NOTTRANSFORMATIONS'.

**Programming Notes:**



## **Subroutine UORIGIN**

### **FUNCTION:**

A user coordinate system is composed at the current beam position. No rotation or scale factors are applied to the new coordinate system.

### **CALLING SEQUENCE:**

**CALL UORIGIN**

### **OPTIONS which may apply:**

'WORKINGAXIS'  
'REFERENCEAXIS'

### **COMMENTS:**

The invocation of this subroutine causes the creation of a new origin (0,0) at the current beam position with respect to the current origin. The rotation factor is assumed to be zero and the scale factors are assumed to be one. For a detailed discussion of the GCS coordinate system faculty, refer to subroutine UCOSYS.

### **Programming Notes:**



### **Subroutine UOUTLN**

#### **FUNCTION:**

This routine draws a line along each edge of the current device display area into which the user's virtual window will be mapped, or around each edge of the entire device plotting surface.

#### **CALLING SEQUENCE:**

**CALL UOUTLN**

#### **OPTIONS which may apply:**

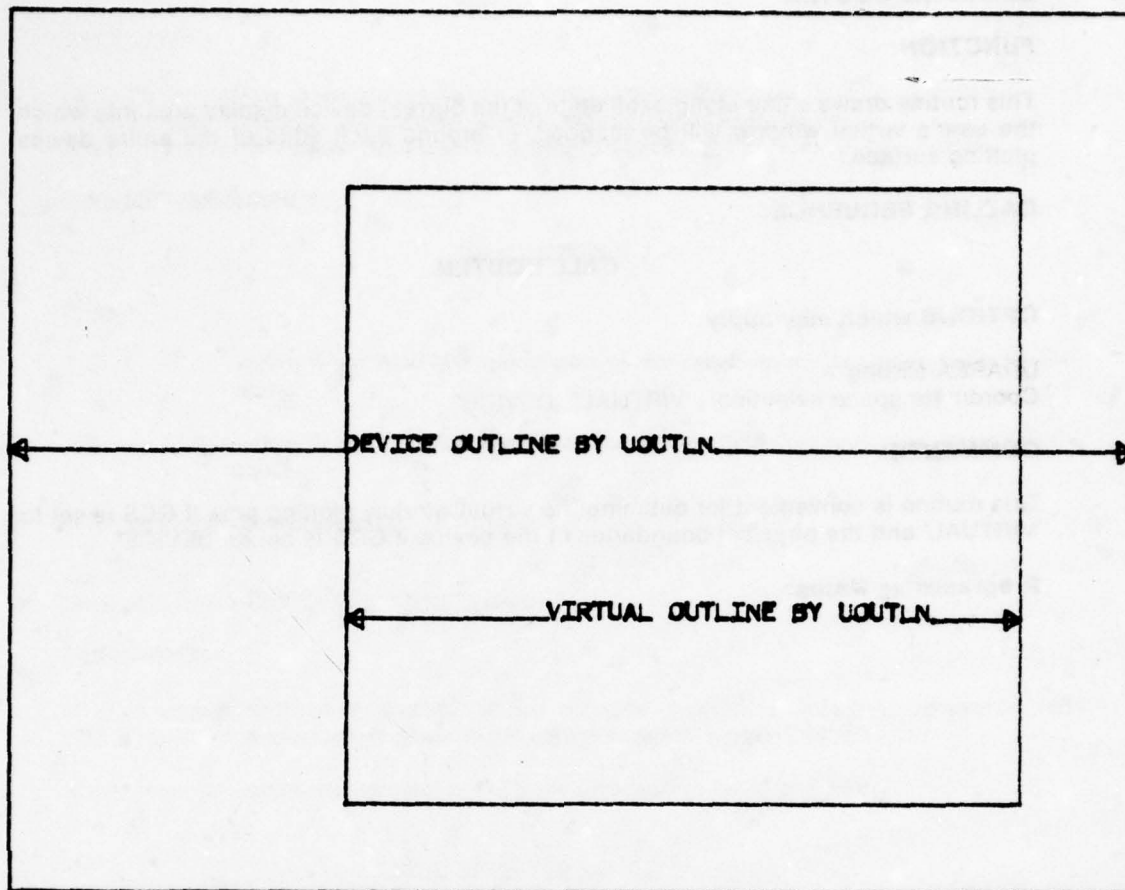
UDAREA setting

Coordinate space selection: 'VIRTUAL', 'DEVICE'

#### **COMMENTS:**

This routine is convenient for outlining the virtual window plotting area if GCS is set to 'VIRTUAL' and the physical boundaries of the device if GCS is set to 'DEVICE'.

#### **Programming Notes:**



```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('EXTRALARGE')
CALL USET ('PERCENT UNITS')
CALL UDAREA (30.,90.,10.,80.)
CALL UOUTLN
CALL USET ('BACKARROW')
CALL UMOVE (0.,30.)
CALL UPEN (30.,30.)
CALL UPRINT (30.,30., 'VIRTUAL OUTLINE BY UOUTLN,')
CALL USET ('ARROW')
CALL UPEN (100.,30.)
CALL USET ('DEVICE')
CALL UOUTLN
CALL USET ('BACKARROW')
CALL UMOVE (0.,50.)
CALL UPEN (30.,50.)
CALL UPRINT (30.,50., 'DEVICE OUTLINE BY UOUTLN,')
CALL USET ('ARROW')
CALL UPEN (100.,50.)
CALL UEND
STOP
END

```

**Subroutine UPAUSE**

**Interactive**

**FUNCTION**

*To suspend execution until one character is entered from keyboard.*

**CALLING SEQUENCE**

**CALL UPAUSE**

**OPTIONS**

*No options apply.*

## **Subroutine UPEN**

### **FUNCTION:**

This routine draws a vector of the type indicated by the current line option from the current beam position (vector tail) to the location specified by the arguments (vector head). The current beam position is updated to point to the head of the vector.

### **CALLING SEQUENCE:**

**CALL UPEN (X,Y)**

Where

**X** is the X- or RADIUS coordinate of the head of the vector in current user units.

**Y** is the Y- or THETA coordinate of the head of the vector in current user units.

### **OPTIONS which may apply:**

See U3PEN

Default Z value - UPSET('ZVALUE',value)

### **COMMENTS:**

All pen movements performed by GCS are made in a three dimensional space. For UPEN, the third dimension is specified by the current X-Y plane, which is zero by default.

*In the three dimensional version of GCS, this plane can be changed by a call to UPSET, i.e.*

**CALL UPSET ('ZVALUE',VALUE)**

*If the current coordinate system is in SPHERICAL coordinates, then VALUE is the PHI component.*

### **Programming Notes:**



### **Subroutine UPEN1**

#### **FUNCTION:**

To draw a vector from the current beam/pen position to the specified endpoint and to set an option which is in effect only during the execution of the subroutine.

#### **CALLING SEQUENCE:**

**CALL UPEN1 (X,Y,OPT)**

Where

**X** is the X- or RADIUS coordinate of the head of the pen/beam movement desired.

**Y** is the Y- or THETA coordinate of the head of the pen/beam movement desired.

**OPT** is the USET option to be set only during the execution of this subroutine.

#### **OPTIONS which may apply:**

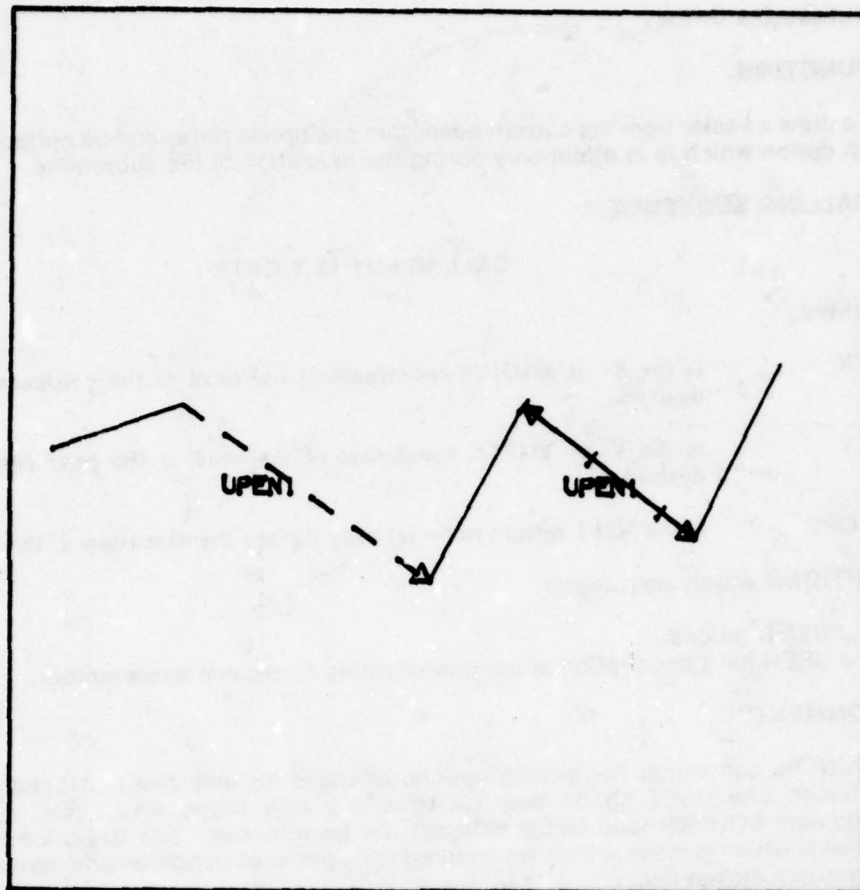
Any USET options

See UPEN for a description of options effecting pen/beam movements.

#### **COMMENTS:**

UPEN1 is convenient for setting options effective for only one pen/beam movement; although any USET option may be specified, only those which may apply to the execution of UPEN (see UPEN writeup) will be effective. The Graphics Status Area variable which is modified will be restored to its previous condition after execution of the beam/pen movement.

#### **Programming Notes:**



```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('EXTRALARGE')
CALL UOUTLN
CALL UMOVE (5.,58.)
CALL UPEN (28.,55.)
CALL UPEN1 (48.,35., 'DARROW')
CALL UPEN (98.,55.)
CALL UPEN1 (88.,48., 'TDOUBLEARROW')
CALL UPEN (98.,68.)
CALL UPRINT (25.,45., 'UPEN1,')
CALL UPRINT (85.,45., 'UPEN1,')
CALL UEND
STOP
END

```

## **Subroutine UPIE**

### **FUNCTION:**

This routine draws a general purpose pie chart. The current user display area will be used and the pie graph will be drawn and scaled according to user specifications.

### **CALLING SEQUENCE:**

**CALL UPIE (ARRAY,PTS,LABELS,XMAXL)**

Where

**ARRAY** is a real array of data points from which the pie chart is drawn  
**PTS** is a real constant or variable which is the number of elements in **ARRAY**  
**LABELS** is an array of labels for the elements within the pie chart.  
**XMAXL** is a real constant or variable which is the maximum length or the length of the largest label in the **LABELS** array.

### **OPTIONS which may apply:**

X Axis Labeling: 'XALPHABETIC' 'NOXLABEL'  
Display Area: 'NEWSCALE' 'OLDSCALE'  
Span: 'SPAN'  
Starting Angle: 'STARTANGLE'

### **COMMENTS:**

UPIE will place the labels from the label array on the individual slices. The specification of 'XALPHABETIC' will cause an X axis label title to be outputted. If the individual slices are not big enough to contain labels then the labels will appear outside of the pie chart. All input data is normalized so that each pie slice appears of a percentage of the total. This subroutine will terminate with error indications if the number of points any input data value, or the maximum pie label size is invalid. It will also terminate if the display area is not of sufficient size.

*The default piechart drawn by a call to UPIE encompasses a full circle with the initial wedge starting at 0. A portion of the circle can be drawn instead by specifying the angle subtended by the graph by calling*

**UPSET ('SPAN',ANGLE).**

*The piechart can be started at an angle other than zero by calling*

**UPSET ('STARTANGLE',ANGLE)**

### **Programming Notes:**



ALGOL - 4%

COBOL - 32%

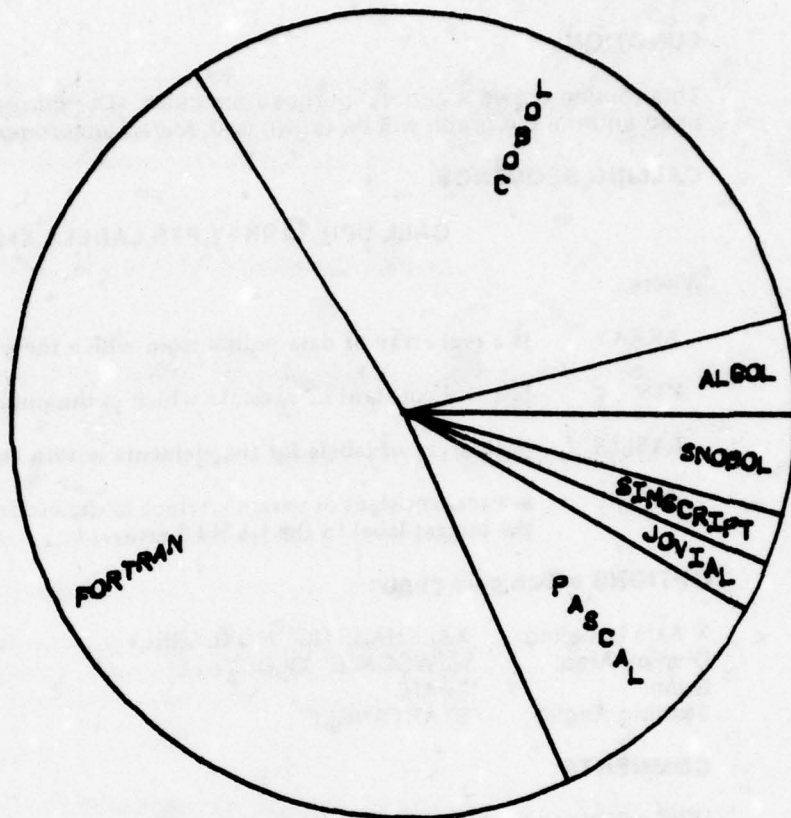
FORTRAN - 46%

PASCAL - 18%

JOVIAL - 2%

SINSCRIPT - 2%

SNOBOL - 4%



TYPICAL LANGUAGE UTILIZATION

```
DIMENSION DATA(7)
CHARACTER LABELS(18(7))
DATA DATA,Y/4.,32.,46.,18.,2.,2.,4.,100./
DATA LABELS/'ALGOL','COBOL','FORTRAN','PASCAL','JOVIAL',
&          'SINSCRIPT','SNOBOL','
CALL USTART
CALL UPSET ('TERMINATOR','')
CALL USET ('XALPHABETIC')
CALL UPSET ('XLABEL','TYPICAL LANGUAGE UTILIZATION,')
CALL UPIE (DATA,7.,LABELS,18.)
CALL USET ('DEVICE')
CALL USET ('PERCENTUNITS')
CALL UDAREA (8.,100.,8.,100.)
DO I I = 1, 7
Y = Y - (100. / FLOAT(7+1))
CALL UMOVE (8.,Y)
CALL UPRTI (LABELS(I),'TEXT')
CALL UPRTI (' - ','TEXT')
CALL UPRTI (DATA(I),'INTEGER')
CALL UPRTI ('X','TEXT')
CONTINUE
CALL UEND
STOP
END
```



**FUNCTION:**

This routine applies a 2-D image translation transformation to the indicated segment.

**CALLING SEQUENCE:**

**CALL UPLACE (X,Y,SEGID)**

Where

**X,Y** is the new position of the segment in current 2-D device units.

**SEGID** is the identifier of a 'retained' segment/frame which was UOPENed for at least 2-D image translations.

**OPTIONS which may apply:**

Device Units: 'INCHES', 'CENTIMETERS', 'FONTUNITS', 'SPECIFICATION UNITS',  
'PERCENT UNITS'

Specification Unit Size (UPSET): 'SPECIFICATION UNITS', 'XSPECIFICATION UNITS',  
'YSPECIFICATION UNITS', 'ZSPECIFICATION UNITS'

Segment Identifier Mode: 'FNAME', 'FNUMBER', 'SNAME', 'SNUMBER'

Segment Type: 'NOTTRANSFORMATIONS', '2DTRANSLATION', '2DGENERAL',  
'3DTRANSLATION', '3DGENERAL'

**COMMENTS:**

The image transformation is applied to the specified segment. If the resulting image exceeds the display dimensions, the result is undetermined.

Image transformations are only applied if supported on the current display surface. Requests for image transformations will be ignored if the display device does not support this facility.

**Programming Notes:**

## **Subroutine UPLLOT**

### **FUNCTION:**

This routine provides a general purpose numeric plotting capability. Given two arrays of corresponding coordinates of one or more curves, it will scale and plot these points along with suitable axes and labels as specified by the user within the current UDAREA. The virtual window will be modified to reflect the resultant scaling.

### **CALLING SEQUENCE:**

**CALL UPLLOT (X,Y,CURVES,PTS,OPTS)**

Where

- X** is the array of X or RADIUS coordinates for the points for all the curves in current user units.
- Y** is an array of Y or THETA coordinates for the points for all the curves in user units.
- CURVES** is a single variable which indicates the number of curves to be plotted.
- PTS** is the array which indicates how many points are in each curve.
- OPTS** is the array which specifies which USET option will apply to each curve as it is being plotted. One option must be specified for each curve and only the first four characters of the option name should be specified.

### **OPTIONS which may apply:**

Coordinate Type Options: 'RECTANGULAR', 'POLAR', 'LOGARITHMIC' UAXIS Options, (see UAXIS)

Scaling Type Options: 'AUTOSCALE', 'FULLSCALE', 'OWNSCALE'

Scale Availability Options: 'NEWSCALE', 'OLDSCALE'

Pen Options: see UPEN

Color Options: see UPEN

Curve Fitting Options: 'NOFITTING', 'FITLINEAR', 'FITPOLYNOMIAL', 'FITSPLINE'

### **COMMENTS:**

UPLLOT is a powerful routine which provides a flexible, yet easy to use means of plotting tabular data. Plotting will take place in 'VIRTUAL' space and the 'SYSTEM' coordinate system (see UCOSYS). The input arrays, X and Y, contain the points for each curve. The number of points in any curve is independent of the number of points in any other curve. This flexibility is possible because the input array POINTS specifies how many points are in each curve. The OPTS array will contain one element for each curve. Each element will contain the first four characters of the name of the USET option which is to apply to the corresponding curve. Normally, these options will be line options (see UPEN). However, they can be any legal USET option. So special effects may be possible by using such options as 'RELATIVE', a color option (see below), or a coordinate type option in conjunction with the use of 'OWNSCALE'. The choice of option is left to the

ingenuity of the user. However, the user is warned that some options may produce ambiguous or indeterminate results.

#### **THE AXES AND LABELS:**

Although there is no restriction as to the number of points in any curve, all points should be of the same coordinate type since this specification will dictate the kind of axes which will appear, and the mode in which the points will be plotted. The axes and labels will be created by a call to the subroutine UAXIS. UAXIS options are available when UAXIS is called from UPLOT. For a description of these options see UAXIS.

#### **THE SCALING:**

Scaling is performed by UAXIS based on its input parameters. If 'NEWSCALE' has been specified, the input parameters will either be calculated by UPLOT from the input arrays, X and Y, ('AUTOSCALE' or 'FULLSCALE') or will be obtained from the current window specification ('OWNSCALE'). If 'OLDSCALE' has been specified the previous scale calculations will be considered active and the scaling factors will not be recalculated.

#### **CURVE FITTING:**

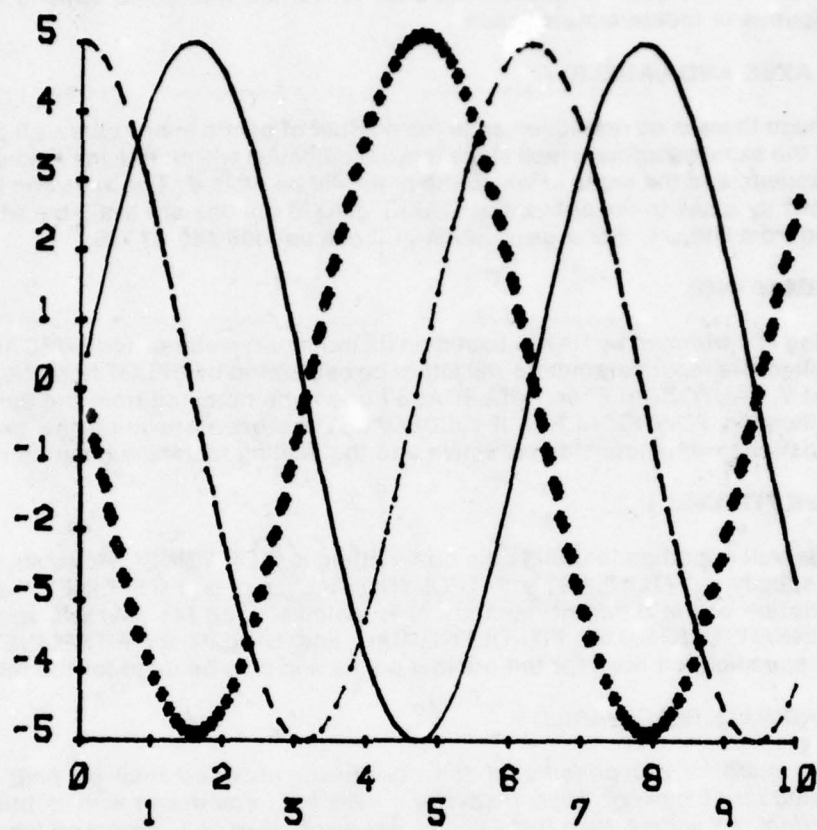
The default condition for automatic curve fitting is 'NOFITTING'. However, the user can also specify a 'FITLINEAR', a 'FITPOLYNOMIAL', a or a 'FITSPLINE'. For a detailed description of the different methods or to calculate the fits yourself, see ULINFT for 'FITLINEAR', ULSTSQ for 'FITPOLYNOMIAL', and 'UPSLIN' for 'FITSPLINE'. The OPTS array specification used for the original points will also be used for the fitted curve.

#### **COORDINATE REPEATABILITY**

*The X and/or Y components of the coordinate provided may be held constant or repeated for all curves. Thus, if several curves are to be drawn with identical X values, but different Y values, then the X values need specified only once, and the USET option 'XREPEAT' specified. If one component is to be held constant, then the USET options 'XCONSTANT' or 'YCONSTANT' may be specified, and only a single X or Y value provided.*

#### **Programming Notes:**





```

DIMENSION X(300), Y(300), POINTS(3)
INTEGER OPTS(3)
DATA OPTS/'LINE','DASH','ANUL'/
DATA POINTS/100.,100.,100./
DO 10 I = 1, 100
  X(I) = FLOAT(I) / 10.
  Y(I) = SIN(X(I)) * 5.0
  Y(I+100) = COS(X(I)) * 5.0
  Y(I+200) = SIN(X(I)+3.14159) * 5.0
10 CONTINUE
CALL USTART
CALL UPSET ('SETDASH',1.)
CALL USET ('PERCENT UNITS')
CALL USET ('SOFT')
CALL UPSET ('HORIZONTAL',.3)
CALL UPSET ('VERTICAL',.5)
CALL USET ('XREPEAT')
CALL UPLT (X,Y,3.,POINTS,OPTS)
CALL UEND
STOP
END

```



### **Subroutine UPLOT1**

#### **FUNCTION:**

This routine provides a general purpose numeric plotting capability. Given two arrays or corresponding coordinates of one curve, it will scale and plot these points along with suitable axes and labels as specified by the user within the current scaling.

#### **CALLING SEQUENCE:**

**CALL UPLOT1 (X,Y,PTS)**

Where

- X** is the array of X- or RADIUS coordinates for the points of the curve in current user units.
- Y** is the array of Y- or THETA coordinates for points of the curve in current user units.
- PTS** is a variable specifying the number of points in the curve.

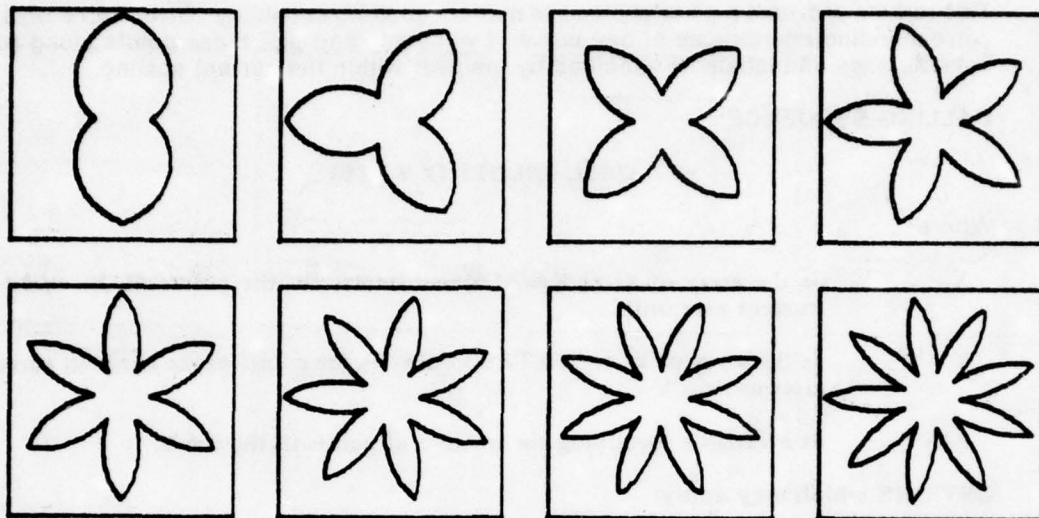
#### **OPTIONS which may apply:**

All UPLOT options

#### **COMMENTS:**

This subroutine will reformat the input information to invoke UPLOT to produce the curve. See UPLOT for a complete description of the options available.

#### **Programming Notes:**



```

DIMENSION R(361),THETA(361)
DATA PI,W,Y0/3.14159265,0.5,4.71/
DO 1 I = 1, 361
1 THETA(I) = FLOAT(I) * PI / 168.8
CALL USTART
CALL USET ('POLAR')
CALL USET ('RADIANS')
CALL USET ('NOAXES')
CALL USET ('NOXLABEL')
CALL USET ('NOYLABEL')
DO 3 I = 1, 2
X0 = -1.5
Y0 = Y0 - 1.6
DO 3 J = 1, 4
W = W + 0.5
X0 = X0 + 1.8
CALL UDAREA (X0,(X0+1.5),Y0,(Y0+1.5))
CALL UOUTLN
DO 2 K = 1, 361
2 R(K) = 1.2-0.7*(ABS(COS(W*THETA(K)))-ABS(SIN(W*THETA(K))))
CALL UPLOT1 (R,THETA,361.0)
3 CONTINUE
CALL UEND
STOP
END

```

### **Subroutine UPLYGN**

#### **FUNCTION:**

This routine creates a regular ploygon of the indicated number of sides and radius centered as specified using the current line option to draw the sides in the currently indicated space.

#### **CALLING SEQUENCE:**

**CALL UPLYGN (X,Y,SIDES,RADIUS)**

Where

**X** is the X- or RADIUS coordinate of the center of the circumscribed circle of the polygon in current user units.

**Y** is the Y- or THETA coordinate of the center of the circumscribed circle of the polygon in current user units.

**SIDES** is the number of sides in the polygon.

**RADIUS** is the radius of the circumscribed circle of the polygon in current user units.

#### **OPTIONS which may apply:**

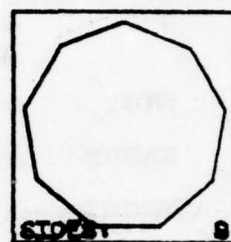
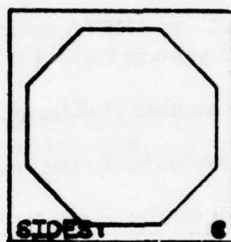
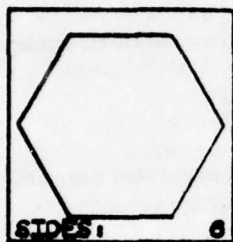
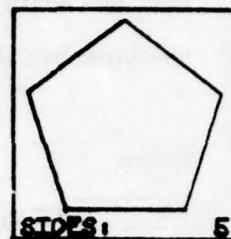
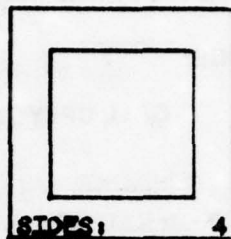
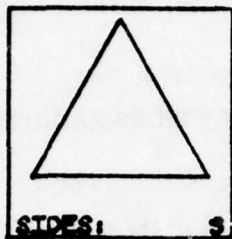
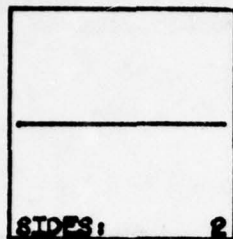
Line Options

Upset Option 'Orientation'

#### **COMMENTS:**

The polygon will be drawn centered at the location specified by the input variables X and Y. It will have a radius (i.e., distance from the center to any point) as indicted by RADIUS, each side of the polygon will be a line of the option currently set in the GSA. The default ORIENTATION of zero degrees will result in the polygon having one side parallel to the current X axis. By specifying an orientation in current angular units, the entire polygon will be displayed rotated appropriately around its center.

#### **Programming Notes:**



```

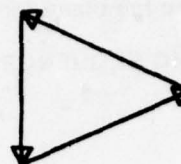
DATA SIDES,Y0/1.0,4.7/
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UWINDO (-1.1,1.1,-1.1,1.1)
DO I I = 1, 2
  X0 = -1.5
  Y0 = Y0 - 1.0
DO J J = 1, 4
  X0 = X0 + 1.0
  SIDES = SIDES + 1.0
CALL UDAREA (X0,(X0+1.0),Y0,(Y0+1.0))
CALL UOUTLN
CALL UPLYGN (0.0,0.0,SIDES,1.0)
CALL USET ('TEXT')
CALL UPRINT (-1.0,-1.05,'SIDES:',')
CALL USET ('INTEGER')
CALL UPRINT (0.0,-1.05,SIDES)
CONTINUE
CALL UEND
STOP
END

```



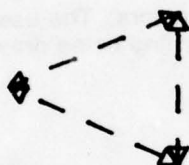


0 DEGREES

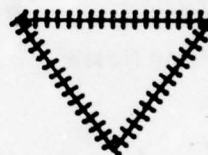


30 DEGREES

UPLYGN WITH VARIOUS LINE OPTIONS AND SHOWING ROTATION OF THE FIGURE



90 DEGREES



60 DEGREES

```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('PERCENT UNITS')
CALL USET ('DEVICE')
CALL UDAREA (10.,40.,60.,100.)
CALL UOUTLN
CALL UPLYGN (25.,75.,3.,10.)
CALL UPRINT (15.,65., '0 DEGREES,')
CALL UDAREA (60.,100.,60.,100.)
CALL USET ('ARROW')
CALL UPSET ('ORIENTATION',30.)
CALL UPLYGN (75.,75.,3.,10.)
CALL UPRINT (65.,60., '30 DEGREES,')
CALL UDAREA (60.,100.,10.,40.)
CALL USET ('TPOINT')
CALL UPSET ('ORIENTATION',60.)
CALL UPLYGN (75.,25.,3.,10.)
CALL UPRINT (65.,10., '60 DEGREES,')
CALL UDAREA (10.,40.,10.,40.)
CALL USET ('DOUBLEARROW')
CALL UPSET ('ORIENTATION',90.)
CALL UPLYGN (25.,25.,3.,10.)
CALL UPRINT (15.,10., '90 DEGREES,')
CALL UPRINT (5.,50., 'UPLYGN WITH VARIOUS LINE OPTIONS AND
& SHOWING ROTATION OF THE FIGURE,')
CALL UEND
STOP
END

```

**FUNCTION:**

This routine defines the point which will be used along with the two end points of the line, to define the plane for line terminators and tic marks.

**CALLING SEQUENCE:**

**CALL UPOINT(X,Y,Z)**

Where

X,Y,Z are the coordinates of some point in current units.

**OPTIONS:**

Pen Coordinate Options: see U3PEN

**COMMENTS:**

The point specified is stored for later use when drawing vectors. The user should be careful to define a point which will not be colinear with any line being drawn.

**Programming Notes:**

**FUNCTION:**

This routine updates the display surface to contain only the defined visible retained segments/frames, deleting all non-retained information.

**CALLING SEQUENCE:**

**CALL UPOST**

**OPTIONS which may apply:**

Security Classification Mode: 'UNSECURED', 'UNCLASSIFIED', 'CONFIDENTIAL',  
'SECRET', 'TOPSECRET'

Segment Visibility Attribute: ('VISIBILITY', 'ON'), ('VISIBILITY', 'OFF')

**COMMENTS:**

All non-retained information will be cleansed from the display surface. This includes remaining segments made invisible while in 'DELAYED' segment posting mode. Non-retained information is considered to be all information generated while not within a 'retained' segment or 'retained' segments where visibility attribute has been set to 'OFF'.

**Programming Notes:**

## **Subroutine UPRINT**

### **FUNCTION:**

This subroutine enables the user to print information at the position specified. The five (5) options available to the user are: 'TEXT', 'REALNUMBER', 'INTEGERNUMBER', 'XYCOORDINATES', and 'XYZCOORDINATES'. The output characters will be either hardware or software depending on the current setting in the Graphics Status Area. Margining will occur with hardware characters and windowing will occur with software characters. The beam will remain positioned at the next character position following the last character sent.

### **CALLING SEQUENCE:**

**CALL UPRINT (X,Y,DATA)**

#### **Where**

- X** is the X- or RADIUS coordinate of the lower left corner of the first character of the output.
- Y** is the Y- or THETA coordinate of the lower left corner of the first character of the output.
- DATA** is a single variable or array containing either real numbers or a GCS text string. The size of data is variable; it is a single variable if 'REALNUMBER' or 'INTEGERNUMBER' is specified; it is a two word array if 'XYCOORDINATES' is specified; it is a three word array if 'XYZCOORDINATES' are specified; and it may be any length if 'TEXT' is specified.

#### **OPTIONS which may apply:**

*All U3PRNT options*

*Default Z value: UPSET ('ZVALUE',VALUE)*

#### **COMMENTS:**

All text functions performed by GCS are performed in a three dimensional space. For UPRINT, the third dimension is defined as the current X-Y plane, which is at Z=0, by default.

*In the three dimensional version of GCS, this plane can be changed by a call to UPSET, i.e.*

**CALL UPSET ('ZVALUE',VALUE)**

If the two dimensional version of GCS is used, then the XY plane corresponds to the screen surface.

#### **Programming Notes:**



### **Subroutine UPRNT1**

#### **FUNCTION:**

This routine displays the data provided at the current beam position in the format currently specified in either hardware or software characters subject to the one time setting of the USET option furnished, which only applies during the executing of this subroutine. Upon return, the beam is positioned at the next character position following the last character produced.

#### **CALLING SEQUENCE:**

**CALL UPRNT1 (DATA,OPTION)**

Where

**DATA** is a single variable or array containing either real numbers or a GCS text string. The size of DATA is variable: it is a single variable if 'REALNUMBER' or 'INTEGERNUMBER' is specified; it is a two word array if 'XYCOORDINATES' is specified; it is a three word array if 'XYZCOORDINATES' is specified; and it may be any length if 'TEXT' is specified.

**OPTION** is a Hollerith-string variable or literal defining the USET option to apply during the execution of this subroutine.

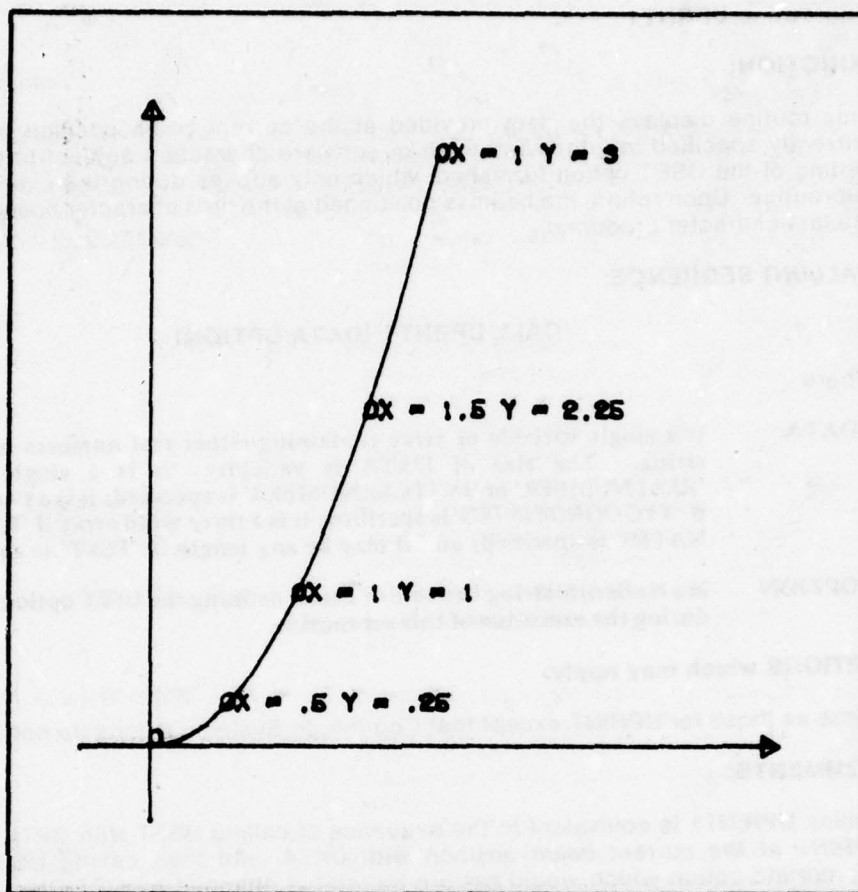
#### **OPTIONS which may apply:**

Same as those for UPRINT except that Coordinate System Options do not apply.

#### **COMMENTS:**

Calling UPRNT1 is equivalent to the sequence of calling USET with OPTIONS, calling UPRINT at the current beam position with DATA, and then calling USET with the appropriate option which would restore the status changed by OPTION. UPRNT1 is convenient for producing strings of output which contain differing data types.

#### **Programming Notes:**



```

CHARACTER OPTION=10(4)
DATA OPTION/'REALNUMBER','INTEGER',
&          'REALNUMBER','INTEGER'/
CHARACTER TITLE#0(2)/' X = ',' Y = ',/
DATA X,Y/0.,0./
CALL USTART
CALL UPSET ('TERMINATOR',',')
CALL UOUTLN
CALL UNINDO (-1.,5.,-1.,5.)
CALL UMOVE (0.,-.5)
CALL UPENI (0.,4.4,'LARROW')
CALL UMOVE (-.5,0.)
CALL UPENI (4.4,0., 'LARROW')
CALL UPENI (X,Y,'NO')
DO 1 I = 1, 4
DO 2 J = 1, 5
X = X + .10
Y = X**2
CALL UPEN (X,Y)
2 CONTINUE
CALL UPENI (X,Y,'NO')
CALL UPRTI (TITLE(1),'TEXT')
CALL UPRTI (X,OPTION(I))
CALL UPRTI (TITLE(2),'TEXT')
CALL UPRTI (Y,OPTION(I))
CALL UMOVE (X,Y)
1 CONTINUE
CALL UEND
STOP
END

```

## **Subroutine UPSET**

### **FUNCTION:**

This routine allows the user to set parameter options in the Graphics Status Area (GSA). Options will remain set to the value specified until changed by a subsequent UPSET or USET call or until the GSA is restored during an UNSAVE or UNSVPN call.

### **CALLING SEQUENCE:**

**CALL UPSET (OPTION,VALUE)**

Where

**OPTION** is the mnemonic option name which uniquely identifies the option to be set or is the first four characters of such an option name.

**VALUE** is the parameter value to be assigned to the option being set. This is always either a REAL number or a character string.

### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

Some UPSET options supplement settings available using USET options. These are included here to allow a means of expansion beyond the discrete USET options. For example, the 'COLOR' UPSET option can be used to set colors not available under USET.

The UPSET options flagged by the notation '3D' indicate that these are present only in the three dimensional version of GCS, and not in the two dimensional version.



## UPSET OPTIONS

Option Name	Value
'ANGLE OF TEXT'	Is an angular value which specifies the angle of the text string in relation of the current X axis. Default value is 0.
'ATTENTION QUEUE SIZE'	An integer indicating the number of words provided in the attention queue. Default is 0.
'ATTITUDE'	Is an angular value which specifies the orientation of the up direction axis with the sides of the window. Default value is 0. Meaning that the up direction is parallel with the left and right window boundaries and pointing towards the top of the window.
'BACKGROUND COLOR'	Is an integer which specifies the color index within the color table for colored backgrounds. Values 0-7 are predefined to represent black, white, red, green, yellow, blue, magenta, and cyan, respectively. Default color is black or none.
'BASE OF LOGARITHMS'	Is a positive value which specifies the base of the logarithms used to perform logarithmic scaling. Default base is 10. This option sets the specified base along each coordinate component.
'BRIGHTNESS'	Is a value between 0 and 100. percent indicating the position in the range of possible line intensity settings from dimmest to brightest. Default brightness is 60%.
'CHARACTER'	Is a Hollerith character which becomes the current system character. The default system character is '*'.
'COLOR'	Is an integer which specifies the color index within the color table. Values 0-7 are predefined to represent black, white, red, green, yellow, blue, magenta, and cyan, respectively. Default color is device-dependent.
'COPY DELAY'	Is a value which indicates the number of seconds of delay required during generation of a hard copy. Default value is device-dependent and is preset to the value required by the selected device.
DESCRIPTOR FILE'	Is a Fortran file number representing the file containing the font descriptors. Default is zero indicating no font file specified.



<b>'DISTANCE'</b>	Is a value measured from the current view plane distance base specifying the position of the view (projection) plane. The default is 0. measured from the view site.
<b>'FONT NAME'</b>	Is a Hollerith string indicating the desired character font. Default is 'GCS' which is the most efficient font.
<b>'GREYSCALE'</b>	Is a value indicating a particular grey level for terminals which support multiple grey scales rather than colors.
<b>'GRID SPECIFICATION'</b>	Is a value containing a dash specification to be used when generating grid axes. Default is 0. which indicates a solid line should be used.
<b>'HORIZONTAL SIZE'</b>	Indicates the width of a software character position in current user units. Default is 5 virtual units.
<b>'INPUT FILE'</b>	Is a Fortran file number indicating which file will be used to obtain graphics input. The default is set to the appropriate computer-system dependent file.
<b>'LABELbROTATION'</b>	Is the number of angular units the axes labels are to be rotated around the axes.
<b>'LIBRARY FILE'</b>	Is a Fortran file number indicating which file should be used by the GCS structure and segmentation facilities as a random work file. Default is 0. indicating no file has been provided.
<b>'LOWER'</b>	Is a Hollerith character which will be used by GCS as the indication to shift to lower case. Default character is '>'.
<b>'MARKER INDEX'</b>	Is an integer value selecting a marker/symbol. Default marker symbol is 0. indicating a point.
<b>'ORIENTATION'</b>	Is an angular value indicating the display orientation of GCS created software symbols and figures such as software characters, polygons, and rectangles. Default orientation is 0.
<b>'OUTPUT FILE'</b>	Is a Fortran file number indicating which file will be used for sending graphics output to the display device. The default is set to the appropriate computer system dependent file.
<b>'POLYNOMIAL DEGREE'</b>	Specifies the degree of the polynomial to be created in calculating a least squares fit through a collection of points. Default value is 5.

'PRECISION'	Specifies the number of significant digits to appear when displaying real numbers. Default value is 4.
'READFILE'	Is a Fortran file number indicating which file will be used to obtain non-graphic input. The default is set to the appropriate computer system dependent file.
'ROTATION'	Same as 'ORIENTATION'.
'SCALEFACTOR'	Specifies a scale to be applied to GCS-created geometric figures such as polygons and rectangles.
'SCRIPTLEVEL'	Is an integer value indicating the scripting level to be set for textual output. Default value is 1.
'SETDASH'	Specifies the characteristics of the dashed lines to be plotted by UPEN. Default value is 56.
'SIZE'	Is a positive integer value which sets hardware character sizes. Values of 1 through 4 correspond to USET options 'SMALL', 'MEDIUM', 'LARGE', and 'EXTRA LARGE' respectively. Default value is 1 for 'SMALL' characters.
'SLANTANGLE'	Is an angular value indicating the amount of slant from the vertical for italicized software characters. Default value is approximately 18 degrees.
'SPAN ANGLE'	Is an angular value indicating the portion of a circle to be occupied by the pie chart. Default value is 360. degrees.
'SPECIFICATION UNITS'	Is a positive value indicating the number of specification units contained in device space for all directions. Default value is 1000.
'SPEED'	Specifies the speed of the communication line in characters per second. Default value is system dependent.
'START ANGLE'	Is an angular value indicating the starting position of the first wedge of the pie chart. Default value is 0.
'STRUCTURE TABLE SIZE'	Is an integer value indicating the number of words in the user provided structure table. Default value is 100.
'SUBSCRIPT CHARACTER'	Is a Hollerith character which will be used to decrease the scripting level by 1. Default subscript character is ' '.
'SUPERScript CHARACTER'	Is a Hollerith character which will be used to increase the scripting level by 1. Default superscript character is ' '.

'SZMARKER'	Is a value in current device units which specifies the size of software generated markers. Default value is device-dependent.
'TABHORIZONTAL'	Is an array of 10 elements containing 10 tab positions in current device units. Default value has all tab stops set to zero.
'TABVERTICAL'	Is an array of 10 elements containing 10 vertical tab positions in current device units. Default value has all tab stops set to zero.
'TERMINATOR'	Is a Hollerith character which will be used as the GCS string terminator character. Default value is ' '.
'TICINTERVAL' 'TICLENGTH'	Specifies the distance in current user units between tic marks of a UPEN created tic line. Default value is 10.
'TICMINUS'	Specifies in current units the size of that portion of a tic mark which lies on the clockwise side of the tic line. Default value is .05 inches.
'TICPLUS'	Specifies in current units the size of that portion of a tic mark which lies on the clockwise side of the tic line. Default value is .05 inches.
'TICX'	Specifies the distance between tic marks or grid lines along X axes. Default value is 0, indicating that a 'nice' number should be chosen.
'TICY'	Is the same as 'TICX' for Y axes.
'TICZ'	Is the same as 'TICX' for Z axes.
'UPPER'	Is a Hollerith character which will be used by GCS as the indication to shift to upper case. Default character is '<'.
'VERTICAL SIZE'	Indicate the height of a software character position in current user units. Default value is seven virtual units.
'WIDTH'	Is a value in current units of the width of a line. Default value is 0, indicating a thin line.
'WRITE FILE'	Is a Fortran file number indicating which file will be used to generate non-graphic output. The default is set to the appropriate computer system dependent file.
'XBASE OF LOGS'	Is a positive value which specifies the base for logarithmic scaling along the X component. Default value is 10.
'XLABEL'	Specifies the alphanumeric label to be displayed along X axes. Default value is 'X '.



'XPERCENT'	Is a value specifying the portion of the width of a software character position to be occupied by a character. Default value is .65 indicating 65% of the width.
'XROTATION'	Is an angular value indicating the amount of rotation around the X axis for UNIVOK structure invocations. Default value is 0.
'XSCALE'	Is the scale factor to be applied along the X axis for UNIVOK structure invocations. Default value is 1.
'XSIZE'	Is the size of hardware or simulated hardware character positions in current device units. Default value is device-dependent and corresponds to 'SMALL' hardware character size.
'XSPECIFICATION UNITS'	Is a positive value indicating the number of X specification units in device space. Default value is 1000.
'YBASE OF LOGS'	Is a positive value indicating the number of X specification units in device space. Default value is 1000.
'YBASE OF LOGS'	Is a positive value which specifies the base for logarithmic scaling along the Y component. Default value is 10.
'YLABEL'	Specifies the alphanumeric label to be displayed along the Y axis. Default value is 'Y'.
'YPERCENT'	Is a value specifying the portion of the height of a software character position to be occupied by a character. Default value is .65 indicating 65% of the height.
'YROTATION'	Is an angular value indicating the amount of rotation around the Y axis for UNIVOK structure invocations. Default value is 0.
'YSCALE'	Is the scale factor to be applied along the Y axis for UNIVOK structure invocations. Default value is 1.
'YSIZE'	Is the size of hardware or simulated hardware character positions in current device units. Default value is device-dependent and corresponds to 'SMALL' hardware character size.
'YSPECIFICATION UNITS'	Is a positive value indicating the number of Y specification units in device space. Default value is 1000.
'ZBASE OF LOGS'	Is a positive value which specifies the base for logarithmic scaling along the Z component. Default value is 10.



**'ZLABEL'**

Specifies the alphanumeric label to be displayed along the Z axis. Default value is 'Z'.

**'ZROTATION'**

Is an angular value indicating the amount of rotation around the Z axis for UINVOK structure invocations. Default value is 0.

**'ZSCALE'**

Is the scale factor to be applied along the Z axis for UINVOK structure invocations. Default value is 1.

**\*means not implemented**

### **Subroutine UQUERY**

#### **FUNCTION:**

This routine obtains the current setting of USET or UPSET options and returns them to the user.

#### **CALLING SEQUENCE:**

**CALL UQUERY(OPTION,VALUE)**

Where

**OPTION** is a character string indicating the USET or UPSET option value desired. Valid options are listed below.

**VALUE** is either an integer array dimensioned to (at least) 10, an integer scalar, or a real scalar, depending on the option queried.

#### **OPTIONS which may apply:**

All USET and UPSET options

#### **COMMENTS:**

VALUE is always a scalar except for axis titles in which case it is an array ten elements in length. VALUE is a HOLLERITH string for all USET UQUERY options. For UPSET options with numeric settings, the value returned is converted to the current corresponding user units.

Those options flagged by the notation '3D' are available only in the three dimensional version of GCS, and not in the two dimensional version.

## UQUERY OPTIONS

Type	Query Name	Returns (Default in Parentheses)
USET	'ABUTTING'	Page abutting mode ('NONABUTTING')
USET	'ACENTERING'	Alphanumeric centering ('NOCENTERING')
USET	'ADJUSTMENT'	3D Axis view adjustment option (plane axis plotted on view port or viewed from current view point) ('XYZVIEW')
USET	'ANGULARUNITS'	Current user angular units ('DEGREES')
USET	'ANGLE OF TEXT'	Current angle of text output (0)
UPSET	'ASPECTRATIO'	3D Display Surface aspect ratio
UPSET	'ATTENTION QUEUE SIZE'	Attention queue size (100)
UPSET	'BACKGROUND COLOR'	Background color index (Black = 0)
USET	'BLENDMODE'	Color Blending mode ('SUBTRACTIVE')
USET	'BLINKRATE'	Blink rate ('NOBLINK')
UPSET	'BRIGHTNESS'	Display intensity (60%)
USET	'BUILD'	3D Structure build mode flag ('NOBUILD')
UPSET	'CHARACTER'	Current system character as Hollerith string (*)
USET	'CLIPPING'	Device space clipping flag ('NOCLIP', 'CLIP' or 'INVERTED')
USET	'COLOR'	Current color index (device dependent)
UPSET	'COPY DELAY'	Copy delay time (device dependent)
UPSET	'CSPACING'	Character spacing mode ('HORIZONTAL')
USET	'CURVE'	Current curve approximation mode ('CONTINUOUS')
UPSET	'DASH'	Numeric value corresponding to current dashline specification (56)
USET	'DESCRIPTION'	Current axis option ('TICAXES')
USET	'DETECTABILITY'	Detectability of new segments ('DESENSITIZED')
USET	'DIMENSION'	3D 2D or 3D coordinate terminator switch ('2DCOORDINATES')
UPSET	'DISTANCE'	3D Distance from viewport to screen plane (150)
USET	'EDIT'	3D Data structure edit mode ('COMPRESS' or 'EXPAND')
USET	'ERRORMODE'	Error presentation (ERROR)
USET	'EXECUTE'	3D Structure building visibility ('EXECUTE')
USET	'EXISTENCE'	Current axis existence option ('XYZAXES')
UPSET	'FACTOR'	Scale factor for GCS created software symbols (1.)
USET	'FITMODE'	Curve fitting mode for autoplotting ('NOFITTING')
USET	'FNAMING MODE'	Frame naming mode ('FNAME')
UPSET	'FNTFILE NUMBER'	Font file number (0)

UPSET	'FONT NAME'		Font name ('GCS')
USET	'FORMAT'		Text number format for numeric labelling ('BESTFORMAT')
USET	'GAPMODE'		Gapped line mode ('UNINTERRUPTED')
UPSET	'GREYSCALE'		Numeric value indicating current grey level (device dependent)
UPSET	'GRID'	3D	Numeric value indicating grid axis type option (0)
USET	'HANDEDNESS'	3D	Left or right handed coordinate system ('RIGHTHANDED')
UPSET	'HARDWARE'		Current hardware character size ('SMALL', device dependent)
UPSET	'HORIZONTAL'		Current horizontal software character position size (5)
UPSET	'INFILE'		Graphics input file designation (computer system dependent)
USET	'INPUT'		Graphics input device medium
USET	'ITALICIZATION'		Italicization mode ('NOITALICS')
UPSET	'LABELANGLE'		Label angle around perpendicular (0)
USET	'LETTERTYPE'		Character type ('HARDWARE')
UPSET	'LEVEL'	3D	Subscript/superscript spacing level (1)
UPSET	'LIBRARY'	3D	Data structure library file mode (0)
UPSET	'LIMIT'	3D	Number of errors before automatic stop. 0 means no limit (0)
USET	'LINEOPTION'		Current line option setting ('LNULL')
UPSET	'LOWERCASE'		Lowercase shift character as Hollerith string ('F')
USET	'LOGARITHMIC'	3D	Axes selected for logarithmic transform (logarithmic transform switch) ('NOLOGSCALING')
USET	'LOGTIME OF APPLICATION'		Time of application of logarithmic scaling ('LOGSYSTEM COORDINATES')
USET	'LOGTYPE'	3D	Flag indicating when logarithmic scaling performed ('LOGSYSTEM', 'LOGUSER', 'LOGOBJECT')
USET	'MAPPINGTYPE'	3D	3D to 2D mapping type ('PERSPECTIVE')
USET	'MENUTYPE'	3D	Menu board type ('STANDARD')
USET	'MERGE'	3D	Structure merge mode switch ('IGNORE')
USET	'MESSAGEDEVICE'		Destination of alphanumeric I/O ('PLOTDEVICE')
USET	'MODE'		Current coordinate mode ('ABSOLUTE')
USET	'NUMERIC'	3D	Numeric labels parallel or perpendicular to axis ('PARALLEL')
USET	'ORDER'	3D	3D coordinate system rotation application order ('ZYX')
UPSET	'ORIENTATION'		Angular orientation of display of GCS created symbols (0)



USET	'ORIGIN'		Forced origin switch for axis scaling ('ORIGIN')
UPSET	'OUTFILE'		File number of graphics output file (computer system dependent)
USET	'OUTPUTDEVICE'		Graphical output destination device ('ALLDEVICES')
USET	'PLANE'	3D	Axis label plane ('XYPLANE')
USET	'PLOTSCALE'		Scale option ('NEWSCALE')
UPSET	'POLYNOMIALDEGREE'		Current degree of polynomial fit for curve fitting (5.)
USET	'POSITION'	3D	Current axes positioning ('EDGEAXES')
UPSET	'PRECISION'		Number of digits of precision to be displayed for real numbers. (4.)
USET	'PROJECTION TYPE'		Type of projection ('PERSEPECTIVE')
UPSET	'READ FILE'		File designator for non-graphic input (computer system dependent)
USET	'REPEAT'	3D	Data structure invocation option ('SINGLE' or 'REPEAT')
USET	'REWIND'	3D	Structure file rewind mode ('REWIND')
USET	'SCALE'		Axis scale option ('AUTOSCALE')
USET	'SCRIPT'	3D	Subscript/superscript control ('NOSCRIPT')
USET	'SECURITY LEVEL'		Control of security banners ('UNSECURED')
USET	'SENSITIVITY'	3D	Light pen sensitivity switch ('DESENSITIZE')
USET	'SIDE'	3D	Side of axes on which labels will appear ('NEGATIVE')
USET	'SIZE'		Hardware character size ('SMALL')
UPSET	'SLANT'	3D	Software character italic slant angle (18. degrees)
USET	'SPACE'		Coordinate space ('VIRTUAL')
UPSET	'SPAN'	3D	Angular span for pie charts (360. degrees)
UPSET	'START'	3D	Pie chart starting angle (0.)
USET	'STOP'		Error stopping control ('NOABORT')
USET	'STORAGEMODE'		Segment/Frame retention mode ('RETAINED')
UPSET	'STRUCTURE LIMIT'		Maximum number of structures which can be defined (100.)
UPSET	'SUBSCRIPTCHARACTER'	3D	Current subscript shift character ( )
UPSET	'SUPERSCRIPTCHARACTER'	3D	Current superscript shift character ( )
UPSET	'SYMBOL INDEX'		Choice of marker (0.)
USET	'SYSTEM'		Current coordinate system ('SYSTEM')
UPSET	'TABHORIZONTAL'		Location of current horizontal tab stop (10.)
UPSET	'TABVERTICAL'		Location of current vertical tab stop (10.)
UPSET	'TERMINATOR'		GCS string termination character
USET	'TEXT'		Textual I/O mode ('TEXT')

UPSET	'TICINTERVAL'		Length of UPEN ticintervals (10.)
UPSET	'TICLENGTH'		Length of UPEN ticintervals (10.)
UPSET	'TICMINUS'		Clockwise tic mark size (0.05)
UPSET	'TICPLUS'		Counter-clockwise tick mark size (0.05)
UPSET	'TICX'		X axis tic interval (0.)
UPSET	'TICY'		Y axis tic interval (0.)
UPSET	'TICZ'	3D	Z axis tic interval (0.)
USET	'TIME'		Time series plotting period ('DATES')
USET	'TYPE'		Type of coordinates ('RECTANGULAR')
USET	'UNIFORM'	3D	High level plotting option ('NONUNIFORM')
USET	'UNITS'		Device space units ('INCHES')
USET	'UPAXIS'	3D	Coordinate system viewport vertical axis ('ZPOSITIVE')
UPSET	'UPPERCASE'		Uppercase shift character as Hollirith string ('A')
USET	'USER'		User coordinate system switch (7.)
UPSET	'VERTICAL'	3D	Vertical software character position switch
USET	'VIEWPORT'		Viewport distance base switch ('SITE')
USET	'VISIBILITY'		Framed/segment creation visibility mode ('VISIBLE')
UPSET	'WIDTH OF LINES'	3D	Width of lines (0.)
USET	'WINDOW'		Window type ('NOWINDOW')
UPSET	'WRITE FILE'		Non-graphic alphanumeric output file (Computer system dependent)
UPSET	'XBASE'		Base of log scaling along x-component (real number or string 'E') (10.)
USET	'XLABEL'		X axis labelling option ('XNUMERICLABEL')
USET	'XLOGARITHMIC'		X axis linearity option
UPSET	'XPERCENTAGE OF CHARACTER SPACE'		Portion of horizontal character space occupied by character (0.65)
USET	'XREPETITION MODE'		X component repetition mode for plotting ('NOXREPEAT')
UPSET	'XROTATION'		Rotation factor around X axis for structure invocation (0.)
UPSET	'XSCALING'		Scaling factor along X axis for structure invocation (1.)
UPSET	'XSPECIFICATION UNIT'		Number of XSPECIFICATION units in device space (1000.)
USET	'XSIZE'		Horizontal hardware character position size (device dependent)
UPSET	'XTITLE'		X axis alphanumeric label ('X')
UPSET	'YBASE'	3D	Base of log scaling along Y-component (real number or string 'E') (10.)

USET	'YLABEL'		Y axis labelling option ('YNUMERICLABELS')
USET	'YLOGARITHMIC'		Y axis linearity option
UPSET	'YPERCENTAGE OF CHARACTER SPACE'		Portion of vertical character space occupied by character (0.65)
USET	'YREPETITION'		Y component repetition made for plotting ('NOYREPEAT')
UPSET	'YROTATION'		Rotation factor around Y axis for structure invocation (0.)
UPSET	'YSCALING'		Scaling factor along Y axis for structure invocation (1.)
UPSET	'YSIZE'		Vertical size for hardware characters (device dependent)
UPSET	'YSPECIFICATIONUNITS'		Number of Y specification units in device space (1000.)
UPSET	'YTITLE'		Y axis alphanumeric label
UPSET	'ZBASE'	3D	Base of log scaling along Z axis (10.)
USET	'ZCLIP'		Hither/yon clipping mode ('NOZCLIPPING')
USET	'ZLABEL'	3D	Z axis label option ('ZNUMERICLABELS')
USET	'ZREPETITION MODE'		Z component repetition mode for plotting ('NOZREPEAT')
UPSET	'ZROTATION'		Rotation factor around Z axis for structure invocation (0.)
UPSET	'ZSCALING'		Scaling factor along Z axis for structure invocation (1.)
UPSET	'ZSPECIFICATIONUNITS'		Number of Z specification units in device space (1000.)
UPSET	'ZTITLE'	3D	Z axis title
UPSET	'ZVALUE'	3D	Numeric default Z-value for 2D coordinates (0.)



## Subroutine URAXIS

### FUNCTION:

This routine draws a scaled and labeled set of axis. The position of the beam/pen when URAXIS is called will be the intersection point of the two axes or their extensions. The axis will be drawn from the minimum to maximum values furnished by the user.

### CALLING SEQUENCE:

**CALL URAXIS (XMIN,XMAX,YMIN,YMAX)**

Where

- |             |  |
|-------------|--|
| <b>XMIN</b> | is the minimum value to be displayed for the X axis in current user units. It also indicates the minimum RADIUS component if polar coordinates are being used. |
| <b>XMAX</b> | is the maximum value to be displayed for the X axis in current user units. It also indicates the maximum RADIUS component if polar coordinates are being used. |
| <b>YMIN</b> | is the minimum value to be displayed for the Y axis in current user units. It also indicates the minimum THETA component if polar coordinates are being used.  |
| <b>YMAX</b> | is the maximum value to be displayed for the Y axis in current user units. It also indicates the maximum THETA component if polar coordinates are being used.  |

### OPTIONS which may apply:

Coordinate System Options: 'RECTANGULAR', 'POLAR', or 'LOGARITHMIC', or 'LOGARITHMIC'

Coordinate Type Options: 'ABSOLUTE' or 'RELATIVE'

Axis Existence Options: 'XYAXES', 'XAXIS', 'YAXIS', or 'NOAXES'

Axis Drawing Options: 'TICAXES', 'GRIDAXES', or 'PLAINAXES'

X Axis Type Options (Rectangular): 'LINXAXIS', 'LOGXAXIS', or 'LNXAXIS'

Y Axis Type Options (Rectangular): 'LINYAXIS', 'LOGYAXIS', or 'LNYAXIS'

X Axis Labeling Options: 'XNUMERICLABEL', 'XALPHANUMERICLABEL', 'XBOTHLABEL', 'NOXLABEL'

Y Axis Labeling Options: 'YNUMERICLABEL', 'YALPHANUMERICLABEL', 'YBOTHLABEL', or 'NOYLABEL'

Numeric Scaling Format Options: 'BESTFORMAT', 'IFORMAT', or 'GFORMAT'

### PARAMETERS which may be set:

- |               |   |
|---------------|---|
| <b>'TICX'</b> | — The interval between tic marks or grid lines on the X axis in current user units (Rectangular, Linear).<br>The interval between radial tic marks or grid lines in current user units (Polar). |
|---------------|---|



- 'TICY'            — The interval between tic marks or grid lines on the Y axis in current user units (Rectangular, Linear).  
                  The interval between axial grid lines in current user units (Polar).
- 'XLABEL'        — The X axis alphabetic label
- 'YLABEL'        — The Y axis alphabetic label
- 'PRECISION'     — The number of digits of precision for the numeric labels

**COMMENTS:**

The default options for URAXIS are the first options listed in each option type. Unless specified by the user, the tic or grid intervals will be determined automatically by URAXIS. The default for PRECISION is four digits of precision.

This routine operates entirely in the current user coordinate system using the user's units. The vertical position of the X axis will be determined by the vertical position of the beam/pen when URAXIS is called. Similarly, the horizontal position of the Y axis will be determined by the horizontal position of the beam/pen. Normally, the numeric and alphabetic labels will be below the X axis and to the left of the Y axis. The X axis labels will move above the X axis if the beam/pen is positioned at or above YMAX, and the Y axis labels will move to the right of the Y axis if the beam/pen is positioned at or to the right of XMAX. The user is cautioned not to position the beam/pen on the boundary of the virtual window, but to leave sufficient room for the labels by positioning the beam/pen well inside the range of the URAXIS arguments, or by making the virtual window boundaries large enough so that room will exist outside the range of the URAXIS arguments.

If POLAR is specified, a check will be made on the arguments to see if a first quadrant or a full four quadrant set of axes is needed. The user should keep this in mind and make sure there is sufficient room for a four quadrant set of axes if needed. If the user is in an ABSOLUTE coordinate system, the axes will be centered at (0,0) regardless of the beam/pen position. If the user is in a RELATIVE coordinate system, the axes will be centered on the beam/pen position. The labels will be to the extreme left and bottom of the axes in both the one quadrant and four quadrant case.

**Programming Notes:**

**FUNCTION:**

This routine enables the user to read character data entered from the alphanumeric keyboard of the terminal. The data will be formatted and returned to the user according to one of five options. The options are: 'TEXT', 'REALNUMBER', 'INTEGER', 'XYCOORDINATES', or 'XYZCOORDINATES' (3D only).

**CALLING SEQUENCE:**

**CALL UREAD (X,Y,DATA,COUNT,FLAG)**

Where

- |              |  |
|--------------|--|
| <b>X</b>     | <b>is the X or RADIUS coordinate of the lower left corner of the point at which the input will begin, if possible.</b>   |
| <b>Y</b>     | <b>is the Y or THETA coordinate of lower left corner of the point at which the input will begin, if possible.</b>  |
| <b>DATA</b>  | <b>is a single variable or array to contain either real numbers or a GCS text string. The size of DATA is variable.</b>  |
| <b>COUNT</b> | <b>is a single variable. If the user has specified 'TEXT' input, then COUNT in the maximum number of characters returned in DATA. The size of DATA must be large enough to contain COUNT characters. The characters are returned in Hollerith format, left justified and blank filled. If the terminal operator inputs fewer characters than requested in COUNT, then DATA will be blank filled to the required size. If the user has specified 'REAL' or 'INTEGER', then the value of COUNT is the number of variables to be input. For 'XYCOORDINATES', the value of COUNT is the number of coordinate pairs to be read. For 'XYZCOORDINATES', the value of COUNT is the number of coordinate triplets to be read.</b> |
| <b>FLAG</b>  | <b>is a single variable. The value of FLAG will be zero if the terminal operator has made an error in entering the numeric data. In this case, the value of DATA is undefined. If the required input was correct, then the value of FLAG is the number of elements in DATA, or the number of characters that the terminal operator entered.</b>  |

**OPTIONS which may apply:**

Coordinate System Options: See UPEN and UCOSYS

ALPHANUMERIC Options: **'TEXT'**, 'INTEGER', 'REALNUMBER', 'XYCOORDINATE'

Device Switching Options: 'MESSAGEDEVICE', 'PLOTDEVICE'

- A. **TEXT** — Under this default option, UREAD will accept and store COUNT characters into DATA: hence, the user must insure that DATA has been suitably dimensioned to hold the number of characters which have been requested. Should fewer than COUNT characters be entered as input, UREAD will store the actual number of characters entered into FLAG, and blank-fill the remaining (COUNT - FLAG) characters of DATA. It should be noted that UREAD does not append the termination character to the end of the input. Therefore, the user is responsible for inserting this character via UAPEND if the string is to be passed as a parameter to UPRINT or UWRITE.

- B. **REALNUMBER** — This option directs UREAD to edit the alphanumeric input as a REAL number, and to store the resulting floating-point number into the single-valued REAL parameter DATA. Should UREAD encounter any illegal characters during the edit, FLAG will be returned with a negative value, and DATA will be undefined.
- C. **INTEGER** — As in the case of REALNUMBER, DATA is assumed to be single-valued and of type REAL. UREAD will edit the alphanumeric input as an INTEGER, perform an INTEGER to REAL conversion, and store the result into DATA. The user may check if the operation was successfully performed by examining FLAG upon return from UREAD.
- D. **XYCOORDINATES** — This option directs UREAD to accept two REAL numbers (separated by a comma) as input and to store them into the two element REAL array, DATA. FLAG is set to reflect the status of the input and editing operation.
- E. **XYZCOORDINATES** — This option directs UREAD to accept three REAL numbers (separated by commas) as input, and store them into the three element REAL array DATA. FLAG is set to reflect the status of the input and editing system.

**COMMENTS:**

The input begins at the current alphanumeric cursor position. For those devices which have alphanumeric capability on the PLOT device, then the alphanumeric cursor position is the current beam position. Otherwise, the input will be from the MESSAGE device. The maximum number of characters retrieved by UREAD is unlimited, however, the number of characters which may be input by the terminal operator may be limited. The user should consult the proper system manual to determine what limits exist, if any, for a particular terminal and system.



### **Subroutine URECT**

#### **FUNCTION:**

This routine draws a rectangle which spans from the current beam position to the diagonally opposite corner.

#### **CALLING SEQUENCE:**

**CALL URECT (X,Y)**

Where

**X** is the X- or RADIUS component of the specified endpoint in current user units.

**Y** is the Y- or THETA component of the specified endpoint in current user units.

#### **OPTIONS which may apply:**

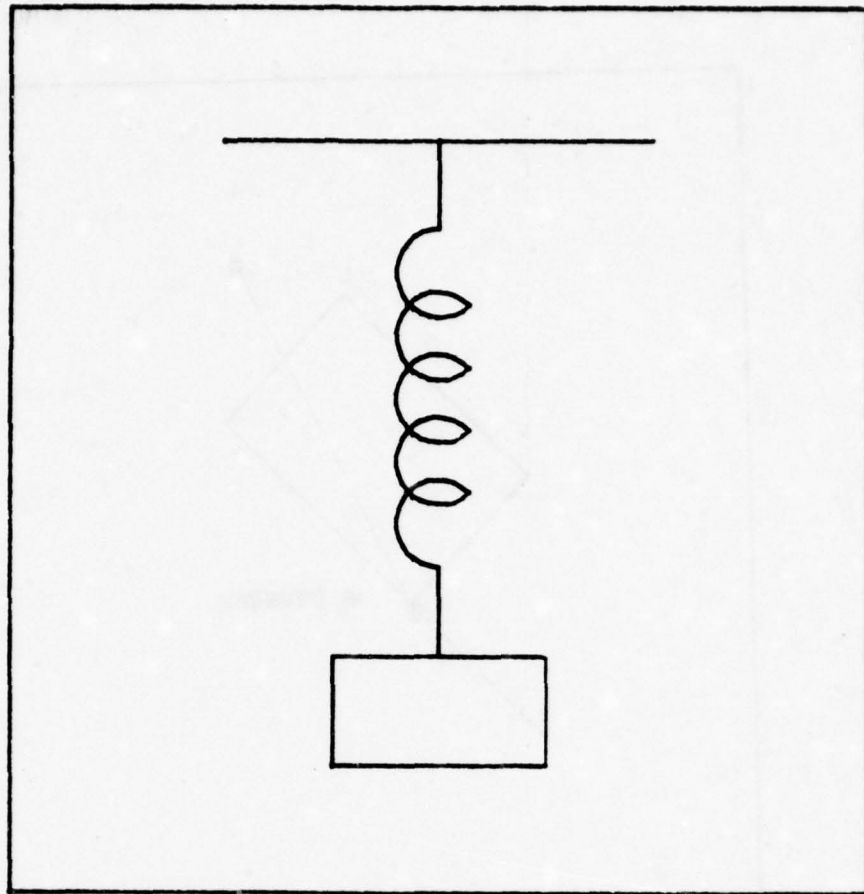
All Line Options  
All Pen Coordinate Options  
UPSET ('ORIENT', ANGLE)

#### **COMMENTS:**

The orientation factor specifies the amount of rotation to be applied to the rectangle. Note that the orientation factor applies to the entire rectangle, and not the endpoint which determines the rectangle.

#### **Programming Notes:**

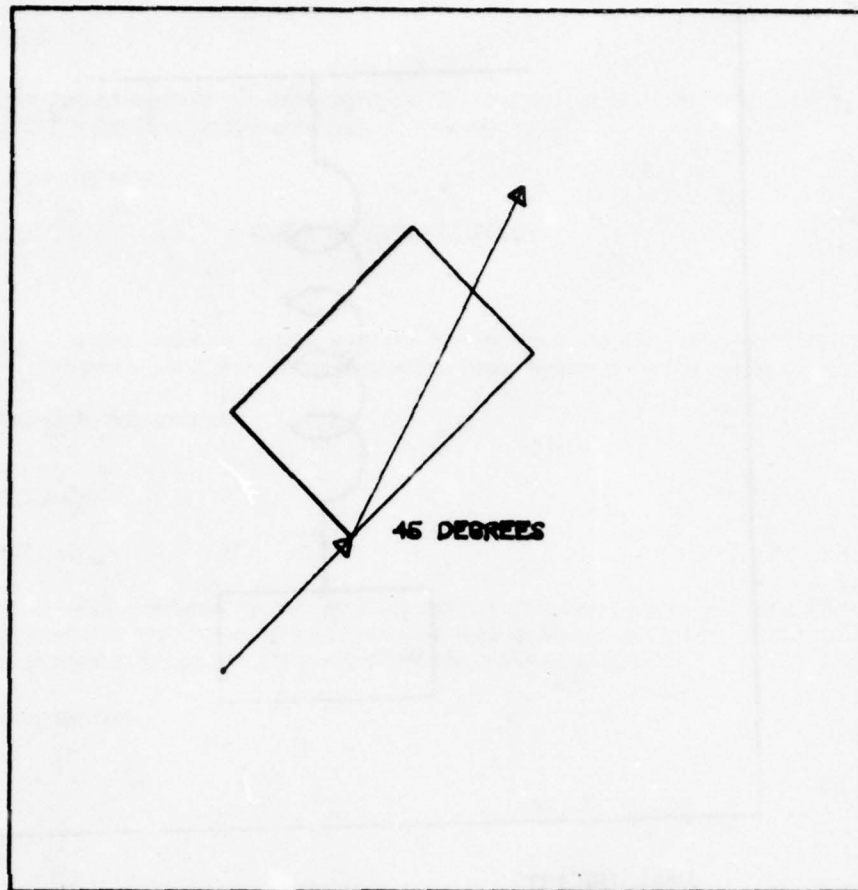




```

CALL USTART
CALL UOUTLN
CALL SPRNAS
CALL UEND
STOP
END
SUBROUTINE SPRNAS
CALL USET ('RELATIVE')
CALL UMOVE (25.0,85.0)
CALL UPEN (50.0,0.0)
CALL UMOVE (-25.0,0.0)
CALL UPEN (0.0,-10.0)
CALL UARC (0.0,-5.0,225.0)
DO I = 1, 3
CALL UARC (-3.535,-3.535,270.0)
1 CONTINUE
CALL UARC (-3.535,-3.535,225.0)
CALL UPEN (0.0,-10.0)
CALL UMOVE (-12.5,-12.5)
CALL URECT (25.0,12.5)
RETURN
END

```



```

CALL USTART
CALL UPSET ('TERMINATOR',',','')
CALL USET ('EXTRALARGE')
CALL UOUTLN
CALL UMOVE (25.,25.)
CALL USET ('ARROW')
CALL UPEN (40.,40.)
CALL UPSET ('ORIENTATION',45.)
CALL USET ('LNULL')
CALL URECT (70.,60.)
CALL USET ('ARROW')
CALL UPEN (60.,60.)
CALL UWRITE (45.,40., '45 DEGREES,')
CALL UEND
STOP
END

```

**Subroutine URESET**

**FUNCTION:**

To reinitialize the Graphics Status Area (GSA) to the default condition.

**CALLING SEQUENCE:**

**CALL URESET**

**OPTIONS which may apply:**

No options apply.

**Programming Notes:**

### **Subroutine UROTAT**

#### **FUNCTION:**

A user coordinate system is composed at the current beam position and rotated according to the rotation factor specified in the argument.

#### **CALLING SEQUENCE:**

**CALL UROTAT (ANGLE)**

Where

**ANGLE** is the rotation factor applied to the new coordinate system which is composed at the current beam position. Angle is in current angular units.

#### **OPTIONS which may apply:**

WORKINGAXIS  
REFERENCEAXIS

#### **COMMENTS:**

The invocation of this subroutine causes the creation of a new origin (0,0) at the current beam position. The unit scale factors are assumed to be equal to one. For a detailed discussion of the GCS coordinate system facility, refer to UCOSYS.

#### **Programming Notes:**



**Subroutine USAREA****FUNCTION:**

This subroutine avoids distortion by changing the device boundaries to maintain a 1 to 1 ratio with the current window boundaries.

**CALLING SEQUENCE:**

**CALL USAREA**

**OPTIONS which may apply:**

No options apply

**COMMENTS:**

The resulting device boundaries can be found by CALL USTUD(ARRAY)

**Programming Notes:**

## **Subroutine USAVE**

### **FUNCTION:**

This routine saves the entire contents of the Graphics Status Area (GSA) in an array furnished by the user. This array will be suitable for later use as input to UNSAVE to restore the status area to the current setting.

### **CALLING SEQUENCE:**

**CALL USAVE (SARRAY)**

Where

**SARRAY** is an array large enough to contain the variables to be saved from the status area.

### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

The size of the array necessary to save the GSA is dependent upon the particular version of GCS being used. For the two dimensional version, SARRAY should be 300 words (decimal) long; for the three dimensional version SARRAY should be 2100 (decimal) words long.

### **Programming Notes:**

**FUNCTION:**

This routine is the basic axis creation routine. It draws a single axis from the designated point for the distance specified parallel to the axis specified.

**CALLING SEQUENCE:**

**CALL USAXIS(AXIS,XSTART,YSTART,ZSTART,DIST)**

Where

**AXIS** specifies the coordinate system axis which is parallel to the desired axis. Its values are as follows:

'XAXIS': axis will be parallel to coordinate system X axis

'YAXIS': axis will be parallel to coordinate system Y axis

'ZAXIS': axis will be parallel to coordinate system Z axis

**XSTART,  
YSTART,  
ZSTART** are the coordinates of the starting point of the axis to be drawn

**DIST** specifies the length of the axis. A positive value will cause the axis to extend in the positive direction. A negative value will cause the axis to extend in the negative direction.

**OPTIONS which may apply:**

X Axis Labeling Options: 'NOXLABELS', 'XNUMERICLABELS',  
'XALPHABETICALABEL', 'XBOTHLABLES'

Y Axis Labeling Options: 'NOYLABELS', 'YNUMERICLABELS',  
'YALPHABETICLABELS', 'YBOTHLABELS'

Z Axis Labeling Options: 'NOZLABELS', 'ZNUMERICLABELS',  
'ZALPHABETICLABELS', 'ZBOTHLABELS'

Alphabetic Label Specification: UPSET('XLABEL', GCS character string)  
UPSET('YLABEL', GCS character string)  
UPSET('ZLABEL', GCS character string)

Numeric Label Format Options: 'BESTFORMAT', 'IFORMAT', 'GFORMAT'

Axis Description Options: 'PLAINAXES', 'TICAXES'

Tic Interval Specifications: UPSET('TICX',value)  
UPSET('TICY',value)  
UPSET('TICZ',value)

Tic Size Specifications: UPSET('TICPLUSSIZE', value)  
UPSET('TICMINUSSIZE', value)

Log Scaling Options: 'NOLOG', 'XLOG', 'YLOG', 'ZLOG', 'XYLOG', 'XZLOG',  
'YZLOG', 'XYZLOG', 'LOGARITHMS'

Log Base Specifications: UPSET('BASE', value)  
UPSET('XBASE', value)

UPSET('YBASE', value)  
UPSET('ZBASE' value)

Axis Label Orientation Options: 'PARALLELABELS', 'PERPENDICULARABELS'

Axis Label Position Options: 'POSITIVESIDE', 'NEGATIVESIDE'

Axis Label Plane Options: 'XYPLANE', 'XZPLANE', 'YZPLANE'

Label Rotation Option: UPSET('LABELROTATION', value)

#### **COMMENTS:**

The axis label options provide for producing numbers which correspond to the tic marks (numeric labels) and for producing a description of the axis (alphabetic labels). These labels are individually selectable for each axis.

The location of the labels can be specified using several different options. The most significant of these is that which specifies the plane in which the characters will be drawn. The three options are 'XYPLANE', 'XZPLANE', and 'YZPLANE'. If the plane selected is perpendicular to the axis being drawn, the angle of rotation around the axis must also be specified by the UPSET ('LABELROTATION', value) call (default is 0 degrees).

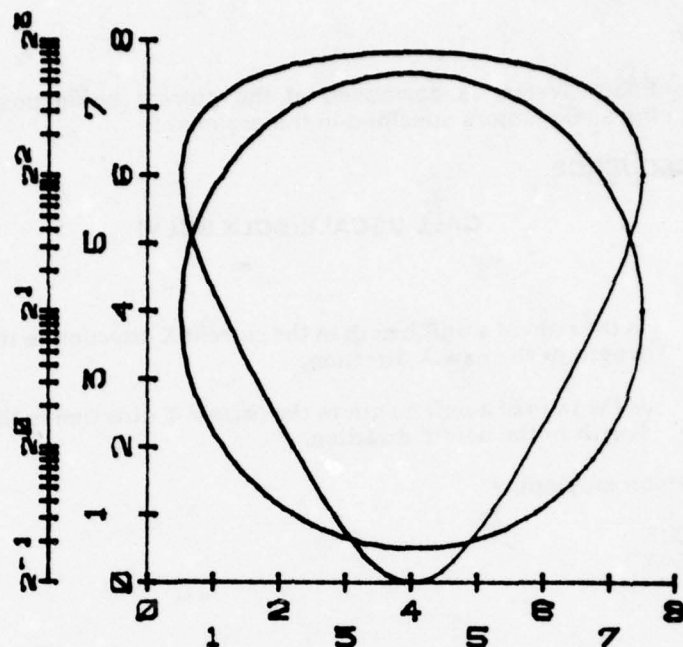
The user can also select which side of the axes the labels will appear. 'POSITIVESIDE' specifies that the labels will be above or to the right of the axis. 'NEGATIVESIDE' specifies that the labels are to the left or below the axis. Default is 'NEGATIVESIDE'.

The orientation of the labels can be specified by either the 'PARALLEL' or 'PERPENDICULAR' UPSET options. PARALLEL specifies that the major axis of the numeric labels (their lengths) will be along the axis. 'PERPENDICULAR' specifies that the major axis of the numeric labels will be perpendicular to the axis. Alpha labels always have their major axis parallel to the axis being drawn. It should be noted that if hardware characters are to be used, vertical spacing may be necessary to maintain the correct label orientation. In some instances, it may even be necessary to produce labels at some oblique angles. Software character labels will, however, always be produced in the proper orientation.

USAXIS will always draw the requested axis in the current coordinate system and space. No attempt is made to insure that the axis is visible. Also no scaling is performed although a tic interval may be calculated if not specified.

#### **Programming Notes:**





LOGGED AND NONLOGGED CIRCLES

```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('SOFTWARE CHARACTERS')
CALL UPSET ('VERTICAL',.4)
CALL UPSET ('HORIZONTAL',.4)
CALL UNINDO (-3.,10.,-3.,10.)
CALL UPSET ('TICX',1.)
CALL UPSET ('TICY',1.)
CALL UPSET ('XLABEL','LOGGED AND NONLOGGED CIRCLES,')
CALL USET ('XBOTHLABELS')
CALL USAXIS ('XAXIS',0.,0.,8.,8.)
CALL USAXIS ('YAXIS',0.,0.,8.,8.)
CALL UCIRCLE (4.,4.,3.5)
CALL USCSYS (0.,2.,0.,1.,2.,1.,0.,0.,0.)
CALL UPSET ('HORIZONTAL',.15)
CALL UPSET ('VERTICAL',.3)
CALL UPSET ('BASE',2.)
CALL USET ('YLOG')
CALL USET ('LOGUSER')
CALL USAXIS ('YAXIS',-1.5,.5,0.,7.5)
CALL UCIRCLE (4.,4.,3.5)
CALL UEND
STOP
END

```

## **Subroutine USCALE**

### **FUNCTION:**

A user coordinate system is composed at the current beam position and scaled according to the scale factors specified in the argument.

### **CALLING SEQUENCE:**

**CALL USCALE(SCLX,SCLY)**

Where

**SCLX** is the ratio of a unit length in the current X direction to the length of a unit length in the new X direction.

**SCLY** is the ratio of a unit length in the current Y direction to the length of a unit length in the new Y direction.

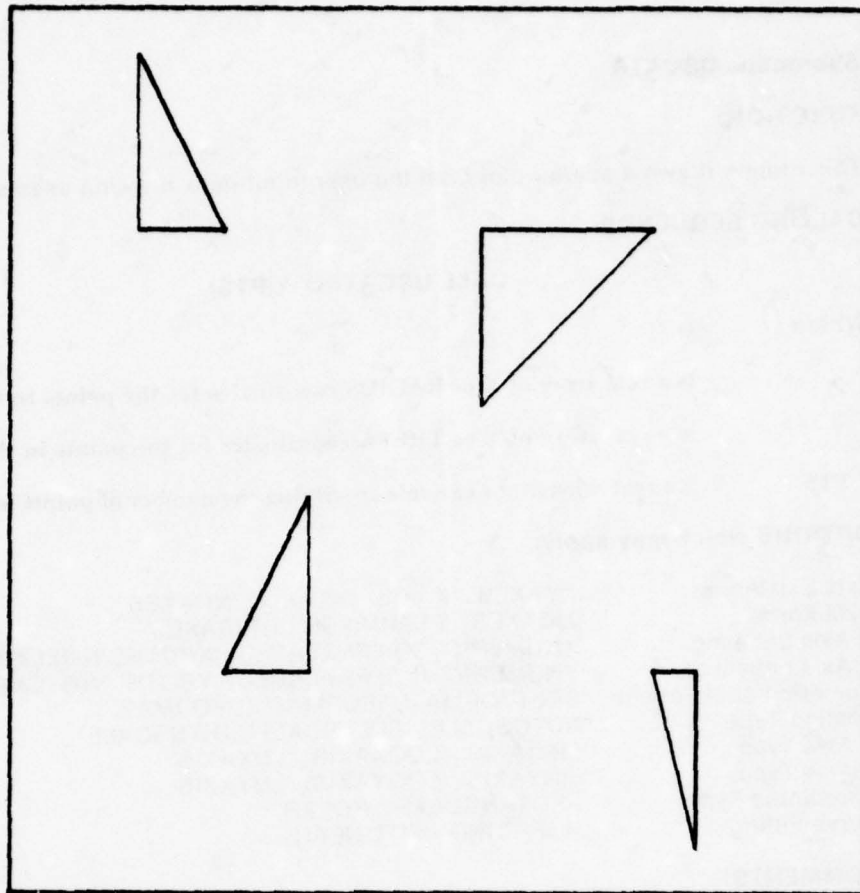
### **OPTIONS which may apply:**

WORKINGAXIS  
REFERENCEAXIS

### **COMMENTS:**

The invocation of this subroutine causes the creation of a new origin (0.,0.) at the current beam position. The rotation factor is assumed to be zero. For a detailed discussion of the GCS coordinate system facility, refer to Subroutine UCOSYS.

### **Programming Notes:**



```

CALL USTART
CALL UOUTLN
CALL UMOVE (25.,25.)
CALL USCALE (1.,1.)
CALL FIGURE
CALL USET ('SYSTEMAXIS')
CALL UMOVE (25.,75.)
CALL USCALE (-1.,1.)
CALL FIGURE
CALL USET ('SYSTEMAXIS')
CALL UMOVE (75.,25.)
CALL USCALE (.5,-1.)
CALL FIGURE
CALL USET ('SYSTEMAXIS')
CALL UMOVE (75.,75.)
CALL USCALE (-2.,-1.)
CALL FIGURE
CALL UEND
STOP
END
SUBROUTINE FIGURE
CALL UPEN (10.,0.)
CALL UPEN (10.,20.)
CALL UPEN (0.,0.)
RETURN
END

```

## **Subroutine USCATR**

### **FUNCTION:**

This routine draws a scatter plot from the user input data, drawing axes as specified.

### **CALLING SEQUENCE:**

**CALL USCATR(X,Y,PTS)**

Where

**X** is a real array of X or RADIUS coordinates for the points in the diagram

**Y** is a real array of Y or THETA coordinates for the points in the diagram

**PTS** is a real constant or variable specifying the number of points in the diagram

### **OPTIONS which may apply:**

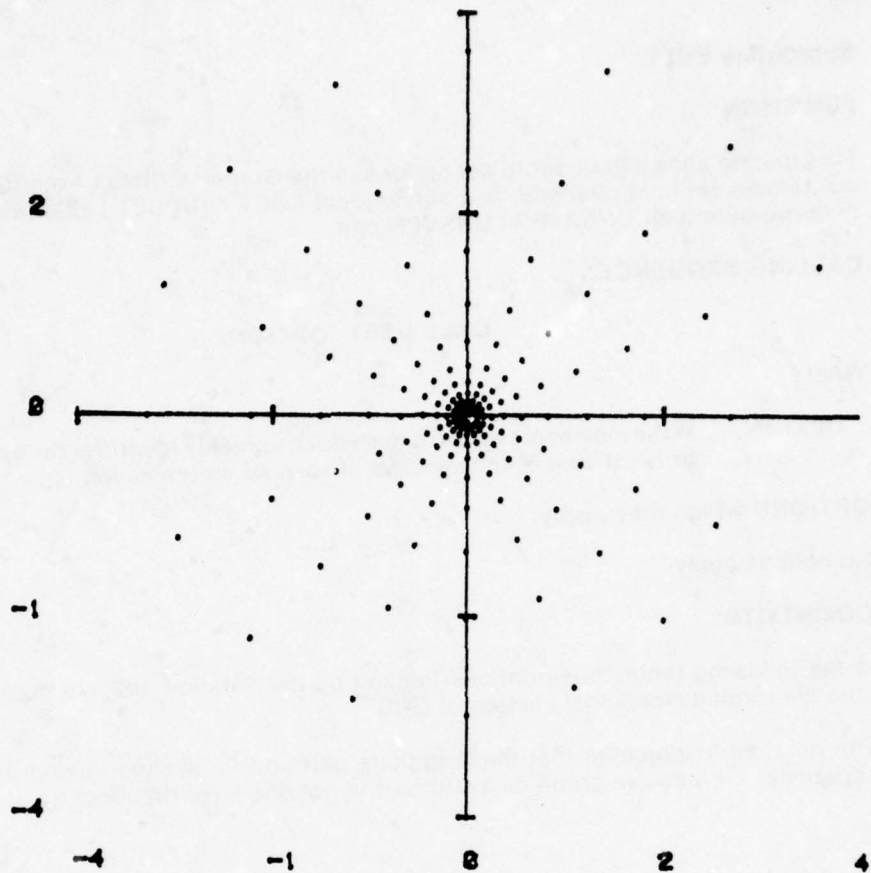
Axis Existence:	'XYAXES', 'XAXIS', 'YXASIX', 'NOAXES'
Axis Format:	'TICAXES', 'PLAINAXES', 'GRIDAXES'
X Axis Labeling:	'XNUMERIC', 'XALPHALABEL', 'XBOTH', 'NOXLABEL'
Y Axis Labeling:	'YNUMERIC', 'YALPHALABEL', 'YBOTH', 'NOYLABEL'
Numeric Label Format:	'BESTFORMAT', 'IFORMAT', 'GFORMAT'
Scaling Type:	'AUTOSCALE', 'FULLSCALE', 'OWNSCALE'
X Axis Type:	'LINXAXIS', 'LOGXAXIS', 'LNXAXIS'
Y Axis Type:	'LINYAXIS', 'LOGYAXIS', 'LNYAXIS'
Coordinate Type:	'RECTANGULAR', 'POLAR'
Curve Fitting:	'NOFITTING', 'FITLINEAR'

### **COMMENTS:**

This routine is very similar to UPLOT with the use of the line option 'NPOINT', i.e., a null line with a point as terminator.

### **Programming Notes:**





```

DIMENSION X(102), Y(102)
DATA PISY6/8.3926998622/
CALL USTART
CALL USET ('POLAR')
CALL USET ('RADIANS')
DO 10 I = 1, 102
  THETA = PISY6 * FLOAT(I)
  X(I) = 4. * EXP(-THETA/10.)
10 Y(I) = THETA
CALL USCATR (X,Y,102.)
CALL UEND
STOP
END

```

## **Subroutine USET**

### **FUNCTION:**

This routine allows the user to set options in the Graphics Status Area (GSA). Options will remain set until changed by a subsequent USET or UPSET call or until the GSA is restored during an UNSAVE or UNSVPN call.

### **CALLING SEQUENCE:**

#### **CALL USET (OPTION)**

Where

**OPTION** is the mnemonic option name which uniquely identifies the option to be set, or is the first four characters of such an option name.

### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

In the following table, those options flagged by the notation '3D' are contained within only the three dimensional version of GCS.

The notation 'I' indicates that these options pertain to a system having an interactive capability. On passive graphics systems, the options have no effect.

**USET OPTIONS BY ALPHABETICAL ORDER:**  
(Default Options are in Bold Type)

Option Name		To Request
'ARROW'		Alphanumeric lines with arrow terminators
'ABACKARROW'		Alphanumeric lines with back arrow terminators
'ABEND'	3D,*	To halt execution when error count exceeds certain limit
'ABORT'	3D	To halt execution when error count exceeds certain limit
'ABSb'		Plotting in an absolute coordinate system
'ABSOLUTE'		Plotting in an absolute coordinate system
'ABUTTING'		Abutting of display surface pages
'ACENTER'		To center character output about given location
'ACHARACTER'		Alphanumeric lines with character terminators
'ACOORDINATE'		Alphanumeric lines with ending coordinates indicated
'ADDITIVE'	*	Additive color blending mode
'ADOUBLEARROW'		Alphanumeric lines with arrowhead terminators
'ALLDISPLAYS'		Routing of graphical output to all devices
'ALPHANUMERIC'		Alphanumeric lines with no terminators
'ALTERNATEDISPLAY'		Routing of graphical output to first alternate device
'ANNUAL'		Time series axis scale in yearly intervals
'ANULL'		Alphanumeric lines with no terminators
'APOINT'		Alphanumeric lines with point terminators
'ARROWLINE'		Solid lines with arrow terminators
'ASYMBOL'		Alphanumeric lines with character terminators
'AUTOSCALE'		Automatic scaling for higher level graphing
'BACKARROWLINE'		Solid lines with back arrow terminators
'BALL'		Track ball graphical input
'BLACK'		Background color to be black
'BBLUE'		Background color to be blue
'BCYAN'		Background color to be cyan
'BESTFORMAT'		Numeric label output in best possible format
'BIHOURLY'		Time series axis scale in two hour intervals
'BLACK'		Switch to pen color black
'BLUE'		Switch to pen color blue
'BMAGENTA'		Background color to be magenta
'BRED'		Background color to be red
'BRIGHT'		Highest possible intensity for output
'BUILD'	3D	Structure building
'BWHITE'		Background color to be white
'BYELLOW'		Background color to be yellow
'CENTIMETERS'		Device space coordinates are in centimeters
'CHARACTER'		Null or invisible lines with character terminators
'CJUSTIFICATION'		Alphanumeric center justification
'COMPRESSED'	3D	Data structure editing option
'CONTINUOUS'		Curved lines be interpreted as one pen operation
'COORDINATES'		UPRINT/UWRITE output in (X,Y) coordinate format
'CURSOR'		Graphic cursor as graphic input device
'CWINDOWING'	3D	Circular windowing
'CYAN'		Switch to pen color cyan
'CYLINDRICAL'		Coordinates are to be of the form (R,THETA,Z), where R is the number of units of radius in the X,Y plane, THETA is the number of angular units

		around the Y axis, and Z is the number of units along the Z axis
'DAILY'		Time series axis scale in daily intervals
'DARROW'		Dashed lines with arrow terminators
'DASH'		Dashed lines with null terminations
'DATE'		Time series axis scale in date series interval
'DBACKARROW'		Dashed lines with arrow terminators
'DCHARACTER'		Dashed lines with character terminators
'DCOORDINATES'		Dashed lines with endpoint coordinates indicated
'DDOUBLEARROW'		Dashed lines with double arrow terminators
'DEFERRED'		Deferred error output until uend is called
'DEGREES'		Angular information be interpreted in degrees
'DESENSITIZE'	I	Disabling of pick sensitivity
'DETECTABLE'		Enabling of pick sensitivity
'DEVICE'		Plotting in device space
'DIGITIZER'		Digitizer is graphics input device
'DIMb'		Lowest possible intensity for output
'DIMENSIONLINE'		Solid lines with arrow terminators
'DISPLAY'		Plotting in device space
'DNULL'		Dashed lines with no terminators
'DOUBLEARROW'		Solid lines with double arrow terminators
'DPOINT'		Dashed lines with point terminators
'DSYMBOL'		Dashed lines with character terminators
'DUMP'		To select dump option
'ECHO'	I	Echo alphanumeric input option
'EDGEAXIS'		X and Y axis labels at edge of graph
'ERROROUTPUT'		Immediate error output
'EXECUTE'	3D	Execution of data structure commands as they are built
'EXPANDED'	3D	Data structure editing option
'EXTENDEDMENU'	I,3D	Extended menuing option
'EXTRALARGE'		Extra large character size
'FAST'		Fast blink rate
'FITFRENCH'		Fit french curve to plotted points
'FITLINEAR'		Fit linear function to plotted lines
'FITPOLYNOMIAL'		Fit least squares polynomial to plotted points
'FITSPLINE'		Fit cubic spline curve to plotted points
'FNUMBERMODE'		Frame identifiers provided as numbers
'FONTUNITS'		To indicate device space plotting in font units
'FULLSCALE'		Full scaling for higher level graphing
'FUNCTIONKEYS'		Function keys is graphics input device
'GAPPED'		Alternate light and dark line output
'GFORMAT'		Numeric label output in FORTRAN real (E or F) format
'GOTHIC'		Gothic character font
'GRADS'	3D	Angular units to be measured in grads
'GREEN'		Switch to pen color green
'GRIDAXIS'		Grid axes for higher level graphing
'GROU'		
'HARDWAREFONT'		Output of hardware generated characters
'HIGHLIGHTED'		Highlighted segments
'HITHER/YONbCLIPPING'		Z axis clipping
'HORIZONTAL'	3D	Alphanumeric output to be printed horizontally
'HOURLY'		Time series axis scale in twenty-four hourly intervals
'IFORMAT'		Numeric label output in integer format
'IGNORE'	3D	Ignore duplicate studies on merge file
'INCHES'		Device space coordinates in inches



'INCREMENTAL'		Plotting in an incremental coordinate system
'INTEGER'		UPRINT/UWRITE output in integer format
'INVISIBLE'	3D	Invisible construction of frames and segments
'ITALICS'		Italic character format
'JOYSTICK'	I	Joystick as graphic input device
'KEYBOARD'	I	Keyboard as (pseudo) graphic input device
'LARGE'		Large character size
'LARROW'		Solid lines with arrow terminators
'LBACKARROW'		Solid lines with back arrow terminators
'LCHARACTER'		Solid lines with character terminators
'LCOORDINATES'		Solid lines with endpoint coordinates indicated
'LDOUBLEARROW'		Solid lines with double arrow terminators
'LEFT'	3D	Left handed coordinate system
'LETTER'		UPRINT/UWRITE output in text format
'LIGHTPEN'	I	Light pen as graphic input device
'LINE'		Solid lines with null terminators
'LINXAXIS'		Linear X axis drawing
'LINYAXIS'		Linear Y axis drawing
'LJUSTIFICATION'		Alphanumeric left justification
'LNULL'		Solid lines with null terminators
'LNXAXIS'		Natural log X axis drawing
'LNYAXIS'		Natural log Y axis drawing
'LOGARITHMIC'		Applies logarithmic transforms to all components
'LOGOBJECT'	3D	Logarithmic transforms are to be applied before any other transformations. (In this mode, log transforms may be applied to angle or radius components of 'CYLINDRICAL', 'POLAR', or 'SPHERICAL' coordinates)
'LOGORIGINALUNITS'		Application of log scaling before conversion to rectangular
'LOGSYSTEM'	3D	Logarithmic to be applied after conversion to 'SYSTEM' coordinates (in this mode, the logarithmic coordinate system axes.)
'LOGUSER'	3D	Logarithmic transforms are to be applied after conversion to 'ABSOLUTE', 'RECTANGULAR', 'USER' coordinates but before conversion to 'SYSTEM' coordinates. (In this mode, the logarithmic scaling will be applied along the current 'USER' coordinate system axes.)
'LOGXAXIS'		Base ten log X axis drawing
'LOGYAXIS'		Base ten log Y axis drawing
'LOWERCASE'		Lower case to be 'TEXT' case
'LPOINT'		Solid lines with point terminators
'LSYMBOL'		Solid lines with character terminators
'MAGENTA'		Switch to pen color magenta
'MEDIUM'		Medium character size
'MESSAGEDEVICE'		Alphanumeric I/O routed to a message device
'MILS'	3D	Angular units to be measured in mils
'MINUTELY'		Time series axis scale in minute intervals
'MONTHLY'		Time series axis scale in monthly intervals
'MOUSE'	I	Analog mouse as graphic input device
'MOVE'		Invisible lines with null terminators
'MULTIPLE'	3D	Multiple data structure invocation
'NARROW'		Invisible lines with arrow terminators
'NBACKARROW'		Invisible lines with back arrow terminators
'NCHARACTER'		Invisible lines with character terminators
'NCOORDINATES'		Invisible lines with double arrow terminators
'NDOUBLEARROW'		Invisible lines with double arrow terminators

<b>'NEGATIVESIDE'</b>	3D	Labels will be to the left or below the axes
<b>'NEWSCALE'</b>		New scale for higher level graphing
<b>'NNULL'</b>		Invisible lines with null terminators
<b>'NOBLINE'</b>		Invisible lines with null terminators
<b>'NOABORT'</b>	3D	Do not terminate if error count exceeds specified limit
<b>'NOAXES'</b>		No axes be drawn for higher level graphing
<b>'NOBLINK'</b>		No blinking to occur
<b>'NOBUILD'</b>	3D	No structure building
<b>'NOCENTER'</b>		Text output starts at given point
<b>'NOCLIP'</b>	3D	No device space clipping performed
<b>'NODUMP'</b>	3D	No dump performed
<b>'NOECHO'</b>	1,3D	No echoing of alphanumeric input
<b>'NOEXECUTE'</b>	3D	No execution of data structure commands as they are built
<b>'NOFIT'</b>		No curve fitting for higher level graphics
<b>'NOHIGHLIGHTING'</b>		No segment highlighting
<b>'NOLINE'</b>		Invisible lines with null terminators
<b>'NOLOGARITHMS'</b>	3D	To remove logarithmic transform application
<b>'NONUNIFORM'</b>	3D	High level plotting option
<b>'NOMARK'</b>		Invisible marks with null terminators
<b>'NONABUTTING'</b>		No abutting of display surface pages
<b>'NONRETAINEDSEGMENTS'</b>		Create segments in non-retained form
<b>'NONUNIFORM'</b>		Nonuniform scaling of higher level grading
<b>'NOORIGIN'</b>	3D	No origin to be forced for 'AUTOSCALE' or 'FULLSCALE' scaling options
<b>'NOREPEAT'</b>	3D	No coordinate repeating for high level plotting
<b>'NOREWIND'</b>		No rewind of structure save files
<b>'NORMALINTENSITY'</b>		Normal intensity for output
<b>'NOSCRIP'</b>	3D	To disable any superscripting or subscripting of text output
<b>'NOSIGNIFICANTZEROS'</b>		Suppression of display of significant zeros
<b>'NOSUPERSCRIP'</b>	3D	Same as 'NOSCRIP'
<b>'NOTRAIL'</b>	3D	Record of which GCS routines are invoked is not listed
<b>'NOWINDOWING'</b>	3D	Disable GCS windowing routine
<b>'NOXLABEL'</b>		No labels are to be drawn for graphing
<b>'NOXREPEAT'</b>	3D	To indicate that a component is provided for every X value in every curve in higher level graphing
<b>'NOYLABEL'</b>		No Y labeling for graphing
<b>'NOYREPEAT'</b>	3D	To indicate that a component is provided for every Y value in every curve in higher level graphing
<b>'NOZCLIPPING'</b>		No hither/yon clipping
<b>'NOZLABELS'</b>	3D	To indicate that no Z labels are to be drawn for higher level graphing
<b>'NOZREPEAT'</b>	3D	To indicate that a component is provided for every Z value in every curve in higher level graphing
<b>'NOBLINE'</b>		Invisible line with null terminator
<b>'NOMARK'</b>		Line type
<b>'NPOINT'</b>		Invisible lines with point terminators
<b>'NSYMBOL'</b>		Invisible lines with character terminators
<b>'OLDSCALE'</b>		Old scale to be used for higher level graphing
<b>'ORIGIN'</b>	3D	An origin to be forced for 'AUTOSCALE' and 'FULLSCALE' scaling options
<b>'ORMODE'</b>	1,3D	Asynchronous event processing option
<b>'ORTHOGRAPHIC'</b>	3D	To specify orthographic projection in which the projection is parallel from all points
<b>'OWNSCALE'</b>		Own scale option for higher level graphing

<b>'PARALLELLABELS'</b>	3D	To specify that the main axis of the numeric labels will be parallel to the axis
<b>'PENAXIS'</b>		Axis intersection at current position for graphing
<b>'PENDOWN'</b>		Solid lines with null terminators
<b>'PENORIGIN'</b>	3D	To force the current pen position to be included in the axis range.
<b>'PENUP'</b>		Invisible lines with null terminators
<b>'PERIODIC'</b>		Time series axis scale in accounting period intervals
<b>'PERCENTUNITS'</b>		Device space coordinates specified in percent units
<b>'PERPENDICULARLABELS'</b>	3D	To specify that the major axis the numeric labels will be perpendicular to the axis
<b>'PERSPECTIVE'</b>	3D	To specify perspective projection in which line length diminishes as the distances from the viewing position become greater
<b>'PIRADIANS'</b>	3D	Angular information be interpreted in PI radians
<b>'PLAINAXIS'</b>		Plain axes to be drawn for high level graphing
<b>'PLOTDEVICE'</b>		Alphanumeric I/O to the plotting device
<b>'POINT'</b>		Invisible lines with point terminators
<b>'POLAR'</b>		Plotting in polar (RHO, THETA) units
<b>'POSITIVSIDE'</b>	3D	Labels will be above or to the right of the axis
<b>'PRIMARYDEVICE'</b>		Routing of graphical output to primary device
<b>'QUARTERLY'</b>		Time series axis scale in quarter year intervals
<b>'RADIANS'</b>		Angular information be interpreted in radians
<b>'RASTERUNITS'</b>		To indicate device space coordinates are specified as is in raster units
<b>'REAL'</b>		UPRINT/UWRITE output in real number format
<b>'RECTANGULAR'</b>		Plotting on the user's reference axis
<b>'REDB'</b>		Switch to pen color red
<b>'REFERENCE'</b>		Plotting on the user's reference axis
<b>'REFRESHEDSEGMENT'</b>		Segments to be retained
<b>'RELB'</b>		Plotting in a relative coordinate system
<b>'RELATIVE'</b>		Plotting in a relative coordinate system
<b>'REPLACE'</b>	3D	Data structure building option
<b>'RETAINEDSEGMENTS'</b>		Segments to be retained structures from merge file
<b>'REWIND'</b>	3D	Data structure file handling command
<b>'RIGHTHAND'</b>	3D	Right handed coordinate system
<b>'RJUSTIFICATION'</b>		Alphanumeric right justification
<b>'RWINDOWING'</b>	3D	Rectangular windowing
<b>'SECONDLY'</b>		Time series axis scale in second intervals
<b>'SECRET'</b>		Security classification secret
<b>'SEGMENTED'</b>		Curved lines be interpreted as multiple pen operations
<b>'SEMIANNUAL'</b>		Time series axis scale in semi annual intervals
<b>'SENSITIZE'</b>	3D	Make graphic segments visible
<b>'SIGNIFICATZEROES'</b>		Display of significant zero
<b>'SIMULATED HARDWARE CHARACTERS'</b>		Output of simulated hardware characters
<b>'SINGLE'</b>	3D	Data structure invocation option
<b>'SITEPOINT'</b>	3D	Viewpoint distance to be measured from the view site
<b>'SLOWBLINK'</b>		Slow blink rate
<b>'SMALL'</b>		Use smallest hardware character size
<b>'SOFTWAREFONT'</b>		Output of software generated characters
<b>'SONICPEN'</b>		Sonic pen is graphics input device
<b>'SPECIFIC'</b>		To specify particular device units instead of percent units



'SPHERICAL'		Coordinates are of the form (R,THETA,PHI) where R is the number of units of radius, THETA is the number of angular units around the Z axis, and PHI is the number of angular units around the X axis
'STANDARDMENU'	I,3D	Menuing option
'SUBSCRIPT'	3D	To specify that the output will be lowered from the specified line of text
'SUPERScript'	3D	To specify that the output will be raised from the specified line of text.
'SUPPRESSERRORS'		Error output be suppressed
'SYMBOL'		Invisible lines with character terminators
'SYSTEMAXIS'		Plotting on the system axis
'TABLET'	I	Analog tablet as graphic input device
'TARROW'		Tic lines with arrow terminators
'TBACKARROW'		Tic lines with back arrow terminators
'TCHARACTER'		Tic lines with character terminators
'TCOORDINATES'		Tic lines with endpoint coordinates indicated
'TDOUBLEARROW'		Tic lines with double arrow terminators
'TEXT'		UPRINT/UWRITE output in text format
'SUBTRACTIVE'		Subtractive color blending mode
'TICAXES'		Tic axes to be drawn for higher level graphing
'TICLINE'		Tic lines with null terminators
'TNULL'		Tic lines with null terminators
'THINLINES'		Line width to be thin
'TPOINT'		Tic lines with point terminators
'TRAIL'	3D	To indicate by an identification number which GCS routine is involved.
'TSYMBOL'		Tic lines with character terminators
'TWELVEHOUR'		Time series axis scale in twelve hour intervals
'TWENTYFOURHOUR'		Time series axis scale in twenty four-hour intervals
'UNCLASSIFIED'		Security classification unclassified
'UNDETECTABLE'		Disabling pick sensitivity
'UNIFORM'	3D	High level graphing system
'UNINTERRUPTED'		Non-gapped line output
'UPPERCASE'		Upper case to be 'TEXT' case
'USER'		Plotting on a user defined axis system
'VERTICAL'	3D	Alphanumeric output to be spaced vertically
'VIEWPOINT'	3D	View port distance to be measured from the view point
'VIRTUAL'		Plotting in virtual space
'VISIBLE'	3D	Visible framed output
'WEEKLY'		Time series axis scale in weekly intervals
'WHITE'		Switch to pen color white
'WIDELINES'		Line width to be wide
'WORKING'		New cumulative user coordinate system
'WORLDCOORDINATESYSTEM'		Coordinate system to be the default axis system
'XABSOLUTE'	3D	To specify the X coordinates with respect to the origin of the current coordinate system for indicated components. Y and Z components are to be specified with respect to the current beam/pen position
'XALPHANUMERIC'		An X axis alphanumeric label
'XAXIS'		The X axis be drawn for high level graphing
'XBOTHLABELS'		X axis alphanumeric and numeric labels
'XCONSTANT'	3D	To indicate that the X component does not vary during the drawing of any curve in higher level graphics



'XEDGEYZEROAXIS'		The X axis at edge of graph
'XLOGARITHMIC'		Logarithmic X and linear Y plotting
'XNEGATIVE'	3D	Negative X axis represents up in 3D graphics
'XNUMERIC'	3D	An X axis numeric label
'XPOSITIVE'	3D	Positive X axis represents up in 3D graphics
'XRELATIVE'	3D	To specify the X coordinates with respect to the current beam/pen position. Y and Z components are to be specified with respect to the origin of the current coordinate system
'XREPEAT'	3D	To indicate that one set of X values is provided which will be reused for every curve in higher level graphing
'XYAXES'		The X and Y axes be drawn for high level graphing
'XYABSOLUTE'	3D	To specify the X and Y coordinates with respect to the origin of the current coordinate system for the indicated components. Z components are to be specified with respect to the current beam/pen position
'XYCOORDINATES'		To specify that the data printed by UPRINT/UWRITE is in the form of an (X,Y) coordinate pair.
'XYLOGARITHMIC'	3D	Logarithmic X and Y plotting, linear Z plotting
'XYPLANE'	3D	Labels are to be drawn in the plane formed by the X and Y axes
'XYRELATIVE'	3D	To specify the X and Y components with respect to the current beam/pen position. Z components are to be specified with respect to the origin of the current coordinate system
'XYVIEW'	3D	To view plane formed by X and Y axes
'XYZAXES'	3D	All three axes are to be drawn for higher level graphing
'XYZCOORDINATES'	3D	Text printed by U3PRNT/U3WRIT to be in form of (X,Y,Z) triplet
'XYZLOG'	3D	Applies logarithmic transforms to all three components
'XYZVIEW'	3D	To view all three axes
'XYZb'	3D	To set the rotation application order as indicated
'XYCOORDINATES'		UPRINT/UWRITE output in (X,Y) coordinate format
'XZABSOLUTE'	3D	To specify the X and Z coordinates with respect to the origin of the current coordinate system for the indicated components. Y components are to be specified with respect to the current beam position
'XZAXES'	3D	The X and Z axes are to be drawn for higher level graphing
'XZEROYEDGEAXIS'		The X axis adjacent to boundary of display area
'XZLOGARITHMIC'	3D	Applies Logarithmic transformation to X and Z components
'XZPLANE'	3D	Label plane to be plane formed by X and Z axis
'XZRELATIVE'	3D	To specify the X and Z components with respect to the current beam/pen position. Z components are to be specified with respect to the origin of the current coordinate system
'XZVIEW'	3D	To view the plane formed by the X and Z axes
'XZYb'	3D	To set the rotation application order as indicated
'YABSOLUTE'	3D	Y coordinates are specified with respect to the origin of the current coordinate system for the indicated units. X and Z components are specified with respect to the current beam/pen position

'YALPHANUMERIC'		Y axis alphabetic label
'YAXIS'		The Y axis to be drawn for high level graphing
'YBOTHLABELS'		Y axis having alphabetic and numeric labels
'YCONSTANT'	3D	To indicate that the Y component does not vary during the drawing of any curve
'YEARLY'		Time series axis scale in yearly intervals
'YEDGEZEROAXIS'		The Y axis at edge of graph
'YELLOW'		Switch to pen color yellow
'YLOGARITHMIC'		Logarithmic Y plotting
'YNEGATIVE'	3D	Negative Y direction represents up in 3D graphics
'YNUMERIC'		Y axis numeric label
'YPOSITIVE'	3D	Positive Y direction represents up in 3D graphics
'YRELATIVE'	3D	Y coordinates are specified with respect to the current beam/pen position for the indicated components X and Z components are to be specified with respect to the origin of the current coordinate system
'YREPEAT'	3D	To indicate that one set of Y valves is provided which will be reused for every curve in higher level graphing
'YXZb'	3D	To set the rotation application order as indicated
'YZABSOLUTE'	3D	Y and Z coordinates are specified with respect to the origin of the current coordinate system for the indicated under X components are specified with respect to the current beam/pen position
'YZAXES'	3D	The Y and Z axes to be drawn for higher level graphing
'YZPLANE'	3D	Label plane to be plane formed by Y and Z axes
'YZERXEDGEAXIS'		The Y axis adjacent to boundary of display area
'YZLOGARITHMIC'	3D	Applies logarithmic transformations to Y and Z components
'YZRELATIVE'	3D	Y and Z coordinates to be specified with respect to the current beam/pen position for the Y and Z components, and the X component to be specified with respect to the current beam/pen position for the Y and Z components, and the X component to be specified with respect to the origin of the current coordinate system
'YZVIEW'	3D	To view the plane formed by the Y and Z axes
'YZXb'	3D	To set the rotation application order as specified
'ZABSOLUTE'	3D	Z coordinates to be specified with respect to the origin of the current coordinate system for the indicated component. X and Y components are to be specified with respect to the current beam/pen position
'ZALPHANUMERIC'	3D	Alphanumeric labels to be drawn for higher level drawing
'ZAXIS'	3D	Z axis is to be drawn for higher level graphing
'ZBOTHLABELS'	3D	Both numeric and alphanumeric labels to be drawn for higher level drawing
'ZCLIP'	3D	Clip in Z direction
'ZCONSTANT'	3D	To indicate that the Z component does not vary during the drawing of any curve
'ZEROAXES'		X and Y axes adjacent to boundary of display area
'ZLOGARITHMIC'	3D	Applies logarithmic transform to Z component
'ZNEGATIVE'	3D	Negative Z axis represents up direction in 3D graphics
'ZNUMERIC'	3D	Numeric Z labels to be drawn for high level graphing

'ZPOSITIVE'	3D	Positive Z axis represents up direction in 3D graphics
'ZRELATIVE'	3D	Z coordinates are to be specified with respect to the current beam/pen position. X and Y components are to be specified with respect to the origin of the current coordinate system
'ZREPEAT'	3D	To indicate that one set of Z values is provided which will be reused for every curve in higher level graphing
'ZXYb'	3D	To set the rotation application order as indicated
'ZYXb'	3D	To set the rotation application order as indicated
'12HOUR'		Twelve hour time axis
'13WEEK'		Thirteen week time axis
'2DCOORDINATES'	3D	To specify A coordinate terminator in which two components are listed. (This option is independent of the text coordinate options of 'XYCOORDINATES' and 'XYZCOORDINATES')
'24HOUR'		Twenty four time axis
'3DCOORDINATE'	3D	To specify a coordinate terminator in which all three components are listed. (This option is independent of the text coordinate option of 'XYCOORDINATES' and 'XYZCOORDINATES')

---

NOTE: b - is a blank or space



## **GCS DEFAULT CONDITIONS**

This section addresses those options which are present in the Graphics Status Area as default options. After a call to Subroutine USTART, the Graphics Compatibility System is set to the default conditions, as indicated. The default options can be divided into two groups: Basic Plotting Options and High Level Plotting Options.

### **I. Default Basic Plotting Options**

Plotting is done in 'RECTANGULAR' and 'ABSOLUTE' coordinates on the 'SYSTEM' coordinate axis. The 'USER' coordinate axes are identical to the system axis. Plotting is done in 'VIRTUAL' space with a virtual window whose limits are from 0.0 to 100.0 in the X direction, and from 0.0 to 100.0 in the Y direction. The virtual window is mapped into a display area which is the largest square area on the device display surface. The right hand edge of the square corresponds to the right edge of the display surface.

Character output in GCS will be, by default, in 'HARDWARE' character format, of type 'GOTHIC' and of 'MEDIUM' size. If 'SOFTWARE' characters are requested via USET then the default horizontal size is five (5.0) virtual units, and the default vertical size is seven (7.0) virtual units.

By default, angular units for angular specifications are in 'DEGREES'. If the user switches to 'DEVICE' space, then all length or distance units (i.e. from UPRINT or UREAD), the data will be assumed to be in 'TEXT' mode. The alphanumeric output defaults to the 'PLOT' device if possible, and any graphic output will go to all devices in a cluster. Graphic input is from the primary input device; the number of digits of precision for numeric output is four (4); and GCS detected errors are signalled as they occur.

A line which is drawn by a subroutine in GCS is considered to consist of two parts; a line type and a line terminator. The line type may be solid (visible), ticced, null (invisible), dashed, or alphanumeric. The line may be terminated by an arrow, a back arrow, double arrows, a character, a point, a symbol, a set of coordinate values, or nothing at all. The default line type in GCS is 'SOLID' with 'NULL' terminators ('LNULL'). If 'TICLINES' are requested via USET option, then the default tic interval is ten (10.0) virtual units. It is the user's responsibility to insure that the tic interval is appropriate if he switches to 'DEVICE' space or alters the default virtual window setting or the default display area setting. If 'DASHLINES' are requested, then the default dash specification (56.) will result in a dashed line which is alternately light and dark, in increments of approximately 0.075 inches. If a 'CHARACTER' line terminator is requested but no character is specified by way of Subroutine UPSET, then the character asterisk (\*) is used. Similarly, the asterisk is used to compose the line type if 'ALPHANUMERIC' lines are requested.

### **II. Default High Level Plotting Options**

For each call to UPLOT or UPLOT1, the data values which represent the curves are examined, and a 'NEWSCALE' is created. UAXIS will be invoked to create 'XYAXES'. The data values will be examined, and zero will always appear somewhere on the axis. Numeric labels only will be output for the X axis and the Y axis. The values which appear at the tic marks on the axes will be 'neat' numbers, and the axes will be positioned at the 'EDGE' of the plot. Both the X axis and the Y axis will be drawn in a linear coordinate space. The axis lines will be ticked. If a time series axis is plotted by invoking Subroutine UTAXIS, the default interval for the X axis will be 'DAILY'. No curves will be fit to the data values, but if 'FITPOLYNOMIAL' is requested, then subroutine UPLOT will attempt to fit a fifth degree polynomial to the data.



### III. Summary

COORDINATE SPACE:	'VIRTUAL'
COORDINATE TYPE:	'ABSOLUTE'
COORDINATE SYSTEM:	'RECTANGULAR'
COORDINATE AXIS:	'SYSTEM'
VIRTUAL WINDOW:	0.0 TO 100.0 X DIRECTION 0.0 TO 100.0 Y DIRECTION
DISPLAY AREA:	Largest square area which is right justified on the display surface of the device.
DEVICE SPACE UNITS:	'INCHES'
ANGULAR UNITS:	'DEGREES'
LINE TYPE:	'LINE' or 'LNULL'
SYSTEM CHARACTER:	asterisk (*)
TIC INTERVAL:	10.0 virtual units
DASH SPECIFICATION:	56.
CHARACTER TYPE:	'HARDWARE'
CHARACTER FONT:	'GOTHIC'
CHARACTER SIZE:	'MEDIUM'
SOFTWARE CHARACTER SIZE:	5.0 virtual units horizontal 7.0 virtual units vertical
INPUT/OUTPUT FORMAT:	'TEXT'
ALPHANUMERIC MARGINS:	Device display surface boundaries.
OUTPUT ROUTE:	'PLOTDEVICE'
OUTPUT DISTRIBUTION:	'ALLDEVICES'
GRAPHIC INPUT:	Primary input device
DIGITS OF PRECISION:	4
ERROR HANDLING:	'IMMEDIATE OUTPUT'
AXIS SCALING:	'AUTOSCALE'
AXIS LABELING:	'XNUMERICLABEL' 'YNUMERICLABEL'
AXIS POSITIONING:	'EDGEAXIS'
AXIS TYPE:	'TICAXIS'
AXIS EXISTENCE:	'XYAXES'
AXIS COORDINATES:	'LINXAXIS' 'LINYAXIS'
TIME SERIES AXIS SCALE:	'DAILY'

### USET OPTIONS BY CLASS

#### Coordinate Type

'ABSOLUTE'	'INCREMENTAL'	'RIGHTHANDED'	'ZABSOLUTE'
'RELATIVE'	'XYRELATIVE'	'YABSOLUTE'	'ZRELATIVE'
'XABSOLUTE'	'XZABSOLUTE'	'YRELATIVE'	
'XRELATIVE'	'XZRELATIVE'	'YZABSOLUTE'	
'XYABSOLUTE'	'LEFTHANDED'	'YZRELATIVE'	

#### Coordinate Type

'RECTANGULAR'	'CYLINDRICAL'	'LOGOBJECT'	'XYZLOGARITHMIC'
'POLAR'	'SPHERICAL'	'LOGSYSTEM'	'XZLOGARITHMIC'
'LOGARITHMIC'	'XLOGARITHMIC'	'LOGUSER'	'YZLOGARITHMIC'
'YLOGARITHMIC'	'XYLOGARITHMIC'	'NOLOGARITHMS'	'ZLOGARITHMIC'

### Coordinate Space

'VIRTUAL'	'SPECIFIC'
'DEVICE'	'DISPLAY'

### Device Space Units

'INCHES'  
'CENTIMETERS'  
'FONTUNITS'  
'PERCENTUNITS'  
'RASTERUNITS'

### Angular Units

'DEGREES'	'GRADS'
'RADIANS'	'MILS'
'PIRADIANS'	

### Frame Composition

'INVISIBLE'  
'VISIBLE'

### Line Type

'LINE'	'POINT'	'LNULL'	'NPOINT'
'DASH'	'TICLINE'	'DNULL'	'TNULL'
'CHARACTER'	'NCHARACTER'	'LBACKARROW'	'NNULL'
'SYMBOL'	'NSYMBOL'	'LCOORDINATE'	'LARROW'
'ALPHANUMERIC'	'ANULL'	'NO LINE'	'LDOUBLEARROW'
'MOVE'	'NOLINE'	'NO MARK'	'BACKARROWLINE'
'ARROWLINE'	'NOMARK'	'DIMENSIONLINE'	'COORDINATELINE'
'DOUBLEARROWLINE'			

Lines that are drawn by GCS are composed on a line type and a line terminator. A large number of line types and terminators are possible. The specification is composed by combining the first letter of the line type with the name of the terminator. The following tables illustrate the possible linetypes and terminators:

### LINE TYPE                      FIRST LETTER

LINE	L
DASH	D
ALPHANUMERIC	A
NULL (INVISIBLE)	N
TICLINE	T

### Line Terminators

NULL (NO TERMINATORS)  
CHARACTER  
SYMBOL  
COORDINATE  
POINT  
ARROW  
BACKARROW  
DOUBLEARROW

**Line Repeatability**

'UNINTERRUPTED'  
'GAPPED'

**Curve Approximation**

'CONTINUOUS'  
'SEGMENTED'

**Intensity**

'DIM'  
'BRIGHT'

**Coordinate Axis**

'SYSTEMAXIS'  
'USERAXIS'

**Coordinate Axis Composition**

'WORKINGAXIS'  
'REFERENCEAXIS'

**Character Format**

'GOTHIC'  
'UPPERCASE'  
'ITALIC'  
'LOWERCASE'

**Character Size**

'MEDIUM'  
'SMALL'  
'LARGE'  
'EXTRALARGE'

**Character Type**

'HARDWARE'  
'SOFTWARE'

**Error Conditions**

'ERROR OUTPUT'	'NOABORT'
'SUPPRESSERRORS'	'ABEND'
'DEFERERRORS'	'ABORT'

**Structure Definition**

'BUILD'  
'NOBUILD'

**Structure Building**

'EXECUTE'  
'NOEXECUTE'

### **Structure File Manipulation**

'APPEND'	'REPLACE'
'IGNORE'	'REWIND'
'MULTIPLE'	'SINGLE'

### **Structure Editing**

'COMPRESSED'  
'EXPANDED'

### **Axis Scaling**

'AUTOSCALE'  
'FULLSCALE'  
'OWNSCALE'

### **Axis Scale Existence**

'NEWSCALE'  
'OLDSCALE'

### **Axis Existence**

'XYAXES'	'NOAXES'
'XAXIS'	'XYZAXES'
'YAXIS'	'XZAXES'
'YZAXES'	'ZAXIS'

### **Axis Positioning**

'EDGEAXIS'	
'ZEROAXIS'	
'PENAXIS'	
'XEDGEYZEROAXIS'	'YEDGEXZEROAXIS'
'XZEROYEDGEAXIS'	'YZEROXEDGEAXIS'

### **Numeric Labels**

'BESTFORMAT'  
'FORMAT'  
'GFORMAT'

### **Label Positioning**

'NEGATIVESIDE'	'PERPENDICULARLABELS'
'PARALLELLABELS'	'POSITIVESIDE'
'XYPLANE'	'XZPLANE'
'YPLANE'	

### **X Axis Type**

'LINAXIS'  
'LOGAXIS'  
'LNAXIS'

### **Y Axis Type**

'LINYAXIS'



'LOGYAXIS'  
'LNYAXIS'

#### **X Axis Labels**

'XNUMERIC'  
'XALPHANUMERIC'  
'XBOTHLABELS'  
'NOXLABEL'

#### **Y Axis Labels**

'YNUMERIC'  
'YALPHANUMERIC'  
'NOYLABEL'  
'YBOTHLABELS'

#### **Z Axis Type**

'ZLOGARITHMIC'

#### **Z Axis Type**

'ZNUMERIC'  
'ZALPHANUMERIC'  
'NOZLABEL'  
'ZBOTHLABELS'

#### **Axis Type**

'TICAXIS'  
'PLAINAXIS'  
'GRIDAXIS'

#### **Three Dimensional Windowing**

'SITEPOINT'  
'VIEWPOINT'

#### **Windowing**

'CWINDOWING'  
'NOWINDOWING'  
'RWINDOWINE'

#### **Text Output**

'ACENTER'  
'NOCENTER'  
'NOSCRIPT'  
'XYZCOORDINATES'  
'XYCOORDINATES'  
'2DCOORDINATES'

'NOSUBSCRIPTING'  
'SUBSCRIPT'  
'SUPERScript'  
'INTEGER'  
'TEXT'  
'3DCOORDINATES'

'UPPERCASE'  
'HORIZONTAL'  
'VERTICAL'  
'REAL'

#### **Origin Inclusion**

'NOORIGIN'  
'ORIGIN'  
'PENORIGIN'

### Device Space Clipping

'NOCLIP'  
'ZCLIP'

### Coordinate Repeat Option

'NOREPEAT'	'NOZREPEAT'	'YCONSTANT'	'ZREPEAT'
'NOXREPEAT'	'XCONSTANT'	'YREPEAT'	
'NOYREPEAT'	'XREPEAT'	'ZCONSTANT'	

### Transformation Type

'ORTHOGONAL'  
'PERSPECTIVE'

### Axis Orientation

'XNEGATIVE'	'YNEGATIVE'	'ZNEGATIVE'
'XPOSITIVE'	'YPOSITIVE'	'ZPOSITIVE'

### Three Dimension Viewing

'XYVIEW'	'XZVIEW'
'XYZVIEW'	'YZVIEW'

### Rotation Application Order

'X'Y'Z'	'Y'Z'X'
'X'Z'Y'	'Z'X'Y'
'Y'X'Z'	'X'Y'X'

### Time Axis Scaling

12Hour  
13Week  
24Hour

**FUNCTION:**

To request that the named frame be placed in show status.

**CALLING SEQUENCE:**

**CALL USHOW (NAME)**

**Where:**

**NAME** is the eight character alphanumeric constant or variable of a currently defined frame.

**OPTIONS which may apply:**

No options apply.

**COMMENTS:**

The execution of the subroutine causes the frame NAME to be activated on the face of the display screen. The frame must have been previously defined by a subroutine UFRAME and subroutine FRENDA pair. The initial status of a frame is show status.

**Programming Notes:**

AD-A063 167

ARMY ENGINEER WATERWAYS EXPERIMENT STATION VICKSBURG MISS, F/G 9/2  
GRAPHICS COMPATIBILITY SYSTEM (GCS). PRIMER ON COMPUTER GRAPHIC--ETC(U)  
1978

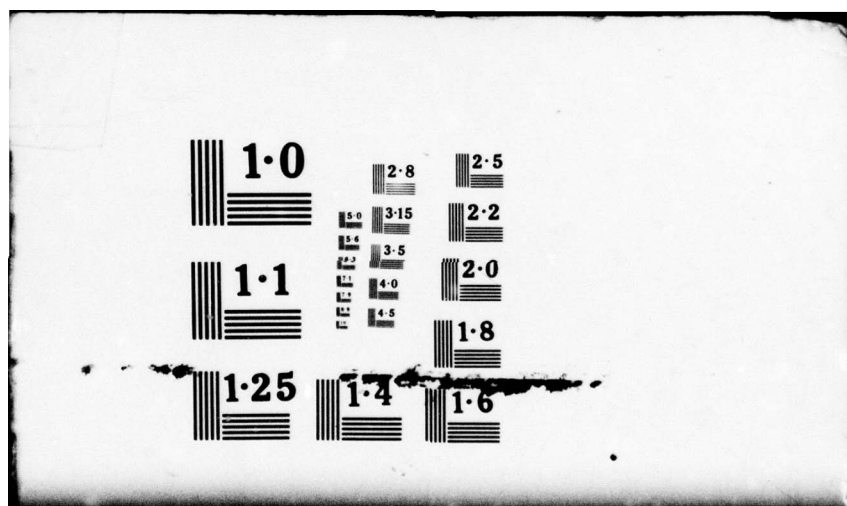
UNCLASSIFIED

NL

5 OF 5  
AD A  
063167







## **Subroutine USPLIN**

### **FUNCTION:**

This routine fits a cubic spline curve to the specified input data and returns a set of X and Y coordinates of data points which lie on the spline curve.

### **CALLING SEQUENCE:**

**CALL USPLIN (X,Y,XN,RX,RY,RN)**

Where

- |           |   |
|-----------|---|
| <b>X</b>  | is an array of XN elements which is the X component of the data points to be fitted   |
| <b>Y</b>  | is an array of XN elements which is the Y component of the data points to be fitted.  |
| <b>XN</b> | is the number of points to be fitted  |
| <b>RX</b> | is an array of RN elements which is the X component of the points which are returned. |
| <b>RY</b> | is an array of RN elements which is the Y component of the points which are returned. |
| <b>RN</b> | is the number of points to be returned (.LE.200).                                     |

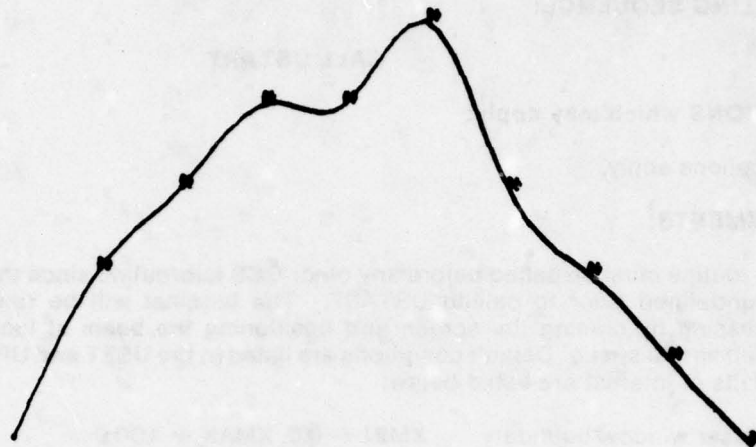
### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

The number of points to be fitted must be greater than two. A maximum of one hundred data points may be fitted. The number of data points to be returned must be at least twice the number of input data points. As the ratio of output points to input points is increased, the resultant spline curve fit appears smoother. All data points are assumed to be in rectangular units.

### **Programming Notes:**



```

DIMENSION X(10),Y(10),RX(50),RY(50)
DATA X/1.,2.,3.,4.,5.,6.,7.,8.,9.,10./
DATA Y/2.,4.,5.,6.,6.,7.,5.,4.,3.,2./
CALL USTART
CALL UINDO (8.,10,5,8.,10,5)
CALL USET ('NCHARACTER')
CALL ULINE (X, Y, 10.)
CALL USPLIN (X, Y, 10.,RX,RY,50.)
CALL USET ('LINE')
CALL ULINE (RX,RY,50.)
CALL UEND
STOP
END

```

## **Subroutine USTART**

### **FUNCTION:**

This routine initializes the Graphics Status Area (GSA) to the default condition and insures that the terminal is ready for graphic output.

### **CALLING SEQUENCE:**

**CALL USTART**

### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

This routine must be called before any other GCS subroutine since the contents of GSA are undefined prior to calling USTART. The terminal will be readied for graphics processing by erasing the screen and positioning the beam at location (0.,0.) in the default virtual space. Default conditions are listed in the USET and UPSET tables. Other defaults of interest are listed below:

- user window boundary - XMIN = 0.0, XMAX = 100.0  
YMIN = 0.0, YMIN = 100.0
- user display area - largest right-justified square that can be defined on the plotting surface.
- margin boundaries - edges of entire display surface.

### **Programming Notes:**



**FUNCTION:**

This routine starts construction of a new graphics data structure. It creates the structure header and places GCS in BUILD mode.

**CALLING SEQUENCE:**

**CALL USTRCT(NAME)**

Where

**NAME** is an eight character data structure name not currently used for this purpose in the current structure file.

**OPTIONS:**

UPSET('LIBRARY',—) must be specified for the structure random file to be defined.

Structure Construction Visibility Options: 'EXECUTE' or 'NOEXECUTE'

**COMMENTS:**

The user again must specify a random file for the data structure library. The structure name must be eight characters, thus blanks must be put in by the user if the name is too short. If the user wishes to display data not to be saved in the structure, he may call USET('NOBUILD') to halt data structure creation. Later on, he may resume construction by calling USET('BUILD'). If the user does not wish to view the construction of his structure, he should specify 'NOEXECUTE'. Otherwise 'EXECUTE' is the default and he will watch his structure being build.

**Programming Notes:**

## **Subroutine USTUD**

### **FUNCTION:**

This routine returns to the user the limits of his display area.

### **CALLING SEQUENCE:**

**CALL USTUD (ARRAY)**

Where

**ARRAY** is an array of at least eight words defined as follows:

**ARRAY(1) = minimum X value in virtual space**  
**ARRAY(2) = maximum X value in virtual space**  
**ARRAY(3) = minimum Y value in virtual space**  
**ARRAY(4) = maximum Y value in virtual space**  
**ARRAY(5) = minimum X value in device space**  
**ARRAY(6) = maximum X value in device space**  
**ARRAY(7) = minimum Y value in device space**  
**ARRAY(8) = maximum Y value in device space**

### **OPTIONS:**

Device Units: 'INCHES', 'CENTIMETERS', 'PERCENT', 'FONT', 'RASTER',  
'SPECIFICATION'

### **COMMENTS:**

A call to this routine will return to the user the limits of his virtual window and display area (virtual) or the limits of the device surface and clip area (device). The device units are in current units.

### **Programming Notes:**

## **Subroutine USVPN**

### **FUNCTION:**

This routine saves the pen-related variables in the Graphics Status Area (GSA) in an array furnished by the user. This array will be suitable for later use as input to UNSVPN to restore the pen-related variables of the status area.

### **CALLING SEQUENCE:**

**CALL USVPN (ARRAY)**

### **Where:**

**ARRAY** is an array large enough to contain the pen-related variables to be saved from the status area.

### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

The size of the array necessary to save the pen-related variables of the GSA is dependent upon the particular version of GCS being used.

Variables to be saved are only those which relate to the functions of the pen. Not saved are those variables which relate to such items as alphanumeric output, high-level graphics, dynamic graphics, and data structure controls. Also not saved are coordinate system specifications which have their own save and restore routines (see USVTR and UNSVTR).

The user is cautioned that any modification to the save array prior to its restoration into the GSA may yield indeterminate results.

### **Programming Notes:**



# **Subroutine USVTR**

## **FUNCTION:**

This routine will save user coordinate system transformation status specification of the Graphics Status Area (GSA).

## **CALLING SEQUENCE:**

**CALL USVTR (SARRAY)**

Where

**SARRAY** is an array of 29 words into which the current coordinate system transform is stored.

## **OPTIONS which may apply:**

No options apply.

## **COMMENTS:**

The invocation of this subroutine will save the current user coordinate system in effect, but will not alter the GCS status.

## **Programming Notes:**



## **Subroutine UTAXIS**

### **FUNCTION:**

This routine creates a set of axes which displays time increments along the horizontal axis and numeric values along the vertical axis. It will support scaling and labeling options as provided for UAXIS. All output from UTAXIS will be contained within the currently defined UDAREA.

### **CALLING SEQUENCE:**

**CALL UTAXIS (BEGPER,PERIOD,YMIN,YMAX)**

#### **Where**

**BEGPER** is the number which represents the beginning time period (1=first period, 2=second period etc.).

**PERIOD** indicates the number of periods to be displayed.

**YMIN** is the minimum Y value to be displayed.

**YMAX** is the maximum Y value to be displayed.

### **OPTIONS which may apply:**

Axis Labeling Options:	'NOXLABELS', 'XNUMERIC', 'XALPHANUMERIC', 'XBOTHLABELS', 'NOYLABELS', 'YNUMERIC', 'YALPHANUMERIC', 'YBOTHLABELS'
Axis Titling Options:	UPSET option 'TICX', UPSET option 'YLABEL'
Tic-Interval Specification:	UPSET option 'XLABEL', UPSET option 'TICY'
Scaling Options:	'AUTOSCALE', 'FULLSCALE', 'OWNSCALE'
Period Options:	'MONTHS', 'YEARS', 'WEEKDAYS', 'DAYS', 'DATE', 'MINUTES', 'HOURS', 'SECONDS', '12 HOUR', '24 HOUR', 'QUARTERS', 'PERIODIC'
Numeric Precision Option:	UPSET option 'PRECISION'

### **COMMENTS:**

This routine will provide a set of axes similar to the UAXIS format. However, at those locations along the X axis where numeric labels would appear, UTAXIS will provide suitable labels to specify those options set in the Graphics Status Area. For a general description of all of the options indicated above except the period options, see UAXIS. Specific comments will be provide below on deviations from those general descriptions.

### **Programming Notes:**

### Period Options:

The period options indicate time divisions which will be used to label the horizontal axis. The time periods are cyclical and will be repeated as necessary until the number of periods specified has been indicated.

'SECONDS'	Continuous from that specified in BEGPER.
'MINUTES'	Continuous from that specified in BEGPER.
'HOURS'	Continuous from that specified in BEGPER.
'DAYS'	Continuous from that specified in BEGPER.
'WEEKDAYS'	Starts with BEGPER; recycles after 'SATURDAY'.
'MONTHS'	Starts with BEGPER; recycles after 'DECEMBER'.
'QUARTERS'	Starts with BEGPER; recycles after '4THQTR'.
'YEARS'	Continuous from that specified in BEGPER.
'DATE'	Starts with BEGPER; recycles on month boundaries. This value should have the format YYYYMMDD or YYMMDD.

### Axis Scaling Options:

The axis scaling option will not be effective for the time period axis. If the axis labels are non-numeric ('WEEKDAYS', 'MONTHS', or 'QUARTERS'), then the first period is assigned value zero and the last period has the value PERIOD-1.

**FUNCTION:**

This routine terminates construction of the current graphics data structure.

**CALLING SEQUENCE:**

**CALL UTERM(NAME)**

Where

**NAME** is the eight character name of the structure currently being built.

**COMMENTS:**

This routine puts a zero word after the last structure element and places GCS in NOBUILD mode.

**Programming Notes:**



**FUNCTION:**

This routine performs standard utility functions for data structure files. Some options will manipulate files, others manipulate structures themselves and their names.

**CALLING SEQUENCE:**

**CALL UTILTY (ACTION,VALUE)**

Where

**ACTION** is a Hollerith character string which specifies one of the following options:

'LOAD'	loads library file from user sequential file
'SAVE'	creates user sequential file from library file
'PURGE'	empties current structure library file
'MERGE'	merges requested file into library file
'DELETE'	deletes structure from library file
'RENAME'	renames old structure to new structure name.

**VALUE** is a Fortran file code number for LOAD,SAVE, and

**MERGE** is a structure name for DELETE, and is an array of two rows containing two structure names for RENAME.

**COMMENTS:**

This routine provides utility manipulations on structures and structure files.

The following comments apply to each separate action.

SAVE writes card image records to a user sequential file of all structures in the library.

LOAD reads the card image user file and puts the information into data structure format in the random file. This action has a compacting effect.

PURGE empties the current library file.

MERGE merges the user card image file into the current structure library. Duplicate names will be REPLACed or IGNORed according to these USET options.

DELETE removes a structure from the library file.

RENAME will rename the structure of the first name given with that of the second name.

**Programming Notes:**



### **Subroutine UTSFIT**

#### **FUNCTION:**

To compute an exponentially smoothed forecast of the input time series data. If desired, the forecast will be trend adjusted.

#### **CALLING SEQUENCE:**

**CALL UTSFIT(ARRAY,POINTS,FCST,ALPHA)**

Where

- ARRAY** is a real array of size **POINTS** which is the input time series data.
- POINTS** is the number of input points.
- FCST** is a real array of size **POINTS** which is the time series forecast for periods 2 to **POINTS** + 1.
- ALPHA** is the weighting factor which determines the amount of reliance placed upon older data. The selection of a large **ALPHA** causes the system to respond rapidly to differences between the actual and forecast data.

#### **OPTIONS which may apply:**

No options apply

#### **COMMENTS:**

The absolute value of **ALPHA** must be less than or equal to one. If the value of **ALPHA** is negative, the forecast will be trend adjusted.

Note that this routine represents one of the basic tools for time series analysis. The user is urged to consult a qualified reference on time series analysis for a complete description of the methods involved in this routine. Other techniques, such as base series correction, cyclic analysis, and higher order smoothing are necessary to synthesize a meaningful, composite forecast. The exponential smoothing of the input data is returned for periods 2...N + 1 where N is the number of input time series periods. The first forecast period which is to be considered accurate must be determined by the user.

#### **Programming Notes:**

**FUNCTION:**

This routine defines the position of the viewer in relation to the environment and specifies his direction of view.

**CALLING SEQUENCE:**

**CALL UVIEW(XVIEW,YVIEW,ZVIEW,XSITE,YSITE,ZSITE)**

Where

**XVIEW,YVIEW,ZVIEW** is the location of the viewing position in current units.

**XSITE,YSITE,ZSITE** is the location of the site at which the viewer is looking in current units.

**OPTIONS:**

Pen Coordinate Options: see U3PEN

Viewing Attitude: UPSET('ATTITUDE',angle)

Up Direction: 'ZPOSITIVE', 'YPOSITIVE', 'XPOSITIVE' 'ZNEGATIVE',  
'YNEGATIVE', 'XNEGATIVE'

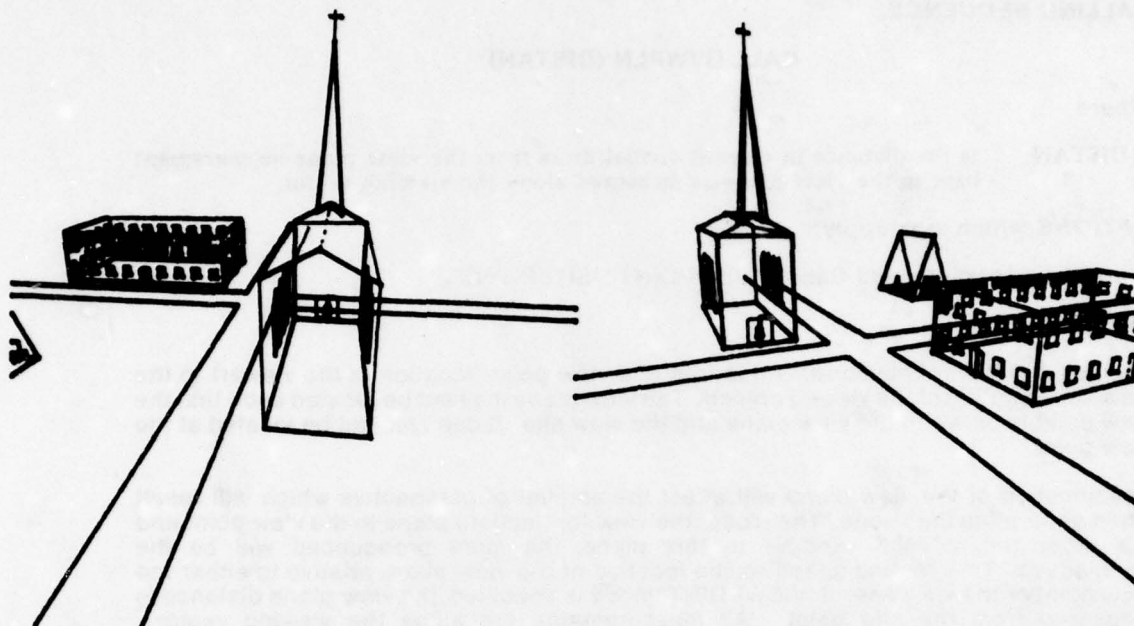
**COMMENTS:**

UVIEW sets the viewing position and direction in current units. Changing the current unit will not effect the viewing position and direction. The attitude of the view is a rotation around the viewing axis (a line drawn from the view point to the view site). It is normally set so that the view will be horizontal. This is the default value (UPSET('ATTITUDE',0)).

Up Direction:

In creating 3-D systems, it has become standard to consider the positive Z axis as representing the direction 'up'. This is the default chosen by GCS. Should the user desire, however, he may specify that any other axis represents the 'up' direction. The 'up' direction refers to the attitude of the viewer in relationship to the coordinate system of the environment.

**Programming Notes:**



```

CALL ATTACH (2, '/TEK3DEND/SAVE, ', 3, 8, 16, )
CALL USTART
CALL UPSET ('TERMINATOR', '<')
CALL USET ('PERCENT UNITS')
CALL USET ('REFERENCE COORDINATE SYSTEM')
CALL UPSET ('LIBRARY', 1.)
CALL UTILITY ('LOAD', 2.)
CALL USET ('VIEWSITE')
CALL UVHPT (150.)
CALL UINDO (-100., 100., -100., 100.)
CALL UDAREA (0., 50., 0., 70.)
CALL UVIEW (-40., 200., 70., -20., 20., 0.)
CALL VILLAG
CALL UVIEW (-70., -150., 50., 0., 0., 10.)
CALL UDAREA (50., 100., 0., 70.)
CALL VILLAG
CALL UEND
STOP
END
SUBROUTINE VILLAG
CALL USET ('XYZ ')
CALL USET ('SYSTEMAXIS')
CALL USET ('REFERENCEAXIS')
CALL USCALL (-50., 20., 0., 1., 1., 1., 90., 0., 0., 'CHURCH')
CALL USCALL (24., -10., 0., 1., 1., 1., 90., -90., 0., 'SCHOOL')
CALL USCALL (70., 70., 0., 0.7, 0.7, 0.7, 0., 0., 0., 'PIZZA')
CALL USCALL (0., 0., 0., 1., 1., 1., 0., 0., 0., 'ROAD ')
RETURN
END

```



**FUNCTION:**

This routine specifies the location of the view (projection) plane along the viewing vector.

**CALLING SEQUENCE:**

**CALL UVWPLN (DISTAN)**

Where

**DISTAN** is the distance in current virtual units from the view plane measurement base to the view plane as measured along the viewing vector.

**OPTIONS which may apply:**

View Plane Measurement Base: 'VIEWPOINT', 'SITEPOINT'

**COMMENTS:**

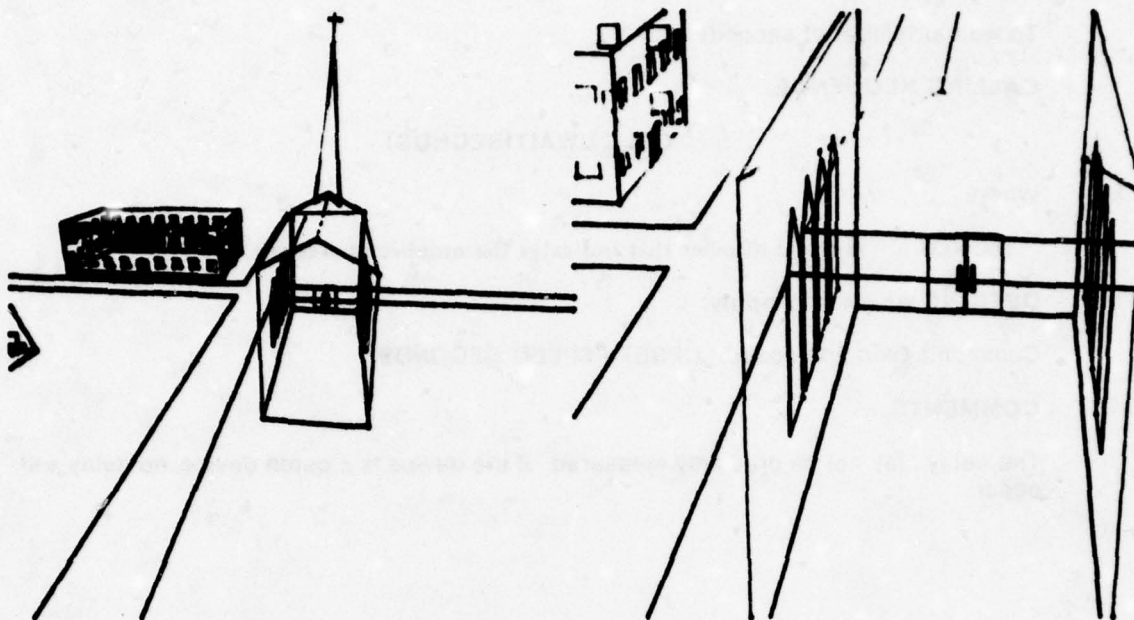
Positive direction of measurement is from the view point (location of the viewer) to the view site (location of the viewed object). The view plane may not be located such that the view point is between the view plane and the view site. It can also not be located at the view point.

Specification of the view plane will effect the amount of perspective which will result when generating the image. The closer the view (projection) plane to the view point and the wider (higher) the window on this plane, the more pronounced will be the perspective. This routine specifies the location of the view plane relative to either the view point or the view site. If 'VIEWPOINT' mode is specified, the view plane distance is measured from the site point. All measurements are along the viewing vector. 'SITEPOINT' is the default mode.

The view plane specification indicates the plane on which the window width and height are measured (see U3WNDO).

**Programming Notes:**





```

CALL ATTACH (2, '/TEKSDENO/SAVE', '3,8,IST,')
CALL USTART
CALL UPSET ('TERMINATOR', '<')
CALL USET ('PERCENT UNITS')
CALL USET ('REFERENCE COORDINATE SYSTEM')
CALL UPSET ('LIBRARY', 1.)
CALL UTILITY ('LOAD', 2.)
CALL USET ('VIEWSITE')
CALL UVWPLN (150.)
CALL UWINDO (-100., 100., -100., 100.)
CALL UDAREA (0., 50., 0., 70.)
CALL UVIEW (-40., 200., 70., -20., 20., 0.)
CALL VILLAS
CALL UVWPLN (500.)
CALL UDAREA (50., 100., 0., 70.)
CALL VILLAS
CALL UEND
STOP
END
SUBROUTINE VILLAS
CALL USET ('XYZ ')
CALL USET ('SYSTEMAXIS')
CALL USET ('REFERENCEAXIS')
CALL USCALL (-50., 20., 0., 1., 1., 1., 00., 0., 0., 'CHURCH')
CALL USCALL (24., -10., 0., 1., 1., 1., 00., -00., 0., 'SCHOOL')
CALL USCALL (70., 70., 0., 0.7, 0.7, 0.7, 0., 0., 0., 'PIZZA')
CALL USCALL (0., 0., 0., 1., 1., 1., 0., 0., 0., 'ROAD ')
RETURN
END

```

**Subroutine UWAIT**

*Interactive*

**FUNCTION:**

To wait a number of seconds.

**CALLING SEQUENCE:**

**CALL UWAIT(SECNDS)**

Where

**SECNDS** is a real number that indicates the number of seconds.

**OPTIONS which may apply:**

Communication line speed: UPSET ('SPEED',SECONDS)

**COMMENTS:**

The delay may not be precisely measured. If the device is a batch device, no delay will occur.

### **Subroutine UWHERE**

#### **FUNCTION:**

This routine places in the arguments the coordinates of the current beam/pen position in current absolute user units.

#### **CALLING SEQUENCE:**

**CALL UWHERE (X,Y)**

#### **Where**

- X** will be on return from this routine, the X- or RADIUS coordinate of the current beam/pen position in current absolute user units.
- Y** will be on return from this routine, the Y- or THETA coordinate of the current beam/pen position in current absolute user units.

#### **OPTIONS which may apply:**

Pen Coordinate options - see UPEN

#### **COMMENTS:**

This routine always returns the coordinates of the current beam position in ABSOLUTE units since RELATIVE units are always (0,0). A handy use for this routine is to obtain the coordinates of the beam/pen in different units from those used to move the beam/pen to where it is located.

#### **Programming Notes:**



## **Subroutine UWINDO**

### **FUNCTION:**

This routine defines the boundaries of the user window into virtual space.

### **CALLING SEQUENCE:**

**CALL UWINDO (XMIN,XMAX,YMIN,YMAX)**

#### **Where**

<b>XMIN</b>	is the left-most window boundary in current VIRTUAL RECTANGULAR units.
<b>XMAX</b>	is the right-most window boundary in current VIRTUAL RECTANGULAR units.
<b>YMIN</b>	is the bottom-most boundary in current VIRTUAL RECTANGULAR units.
<b>YMAX</b>	is the top-most window boundary in current VIRTUAL RECTANGULAR units.

### **OPTIONS which may apply:**

No options apply.

### **COMMENTS:**

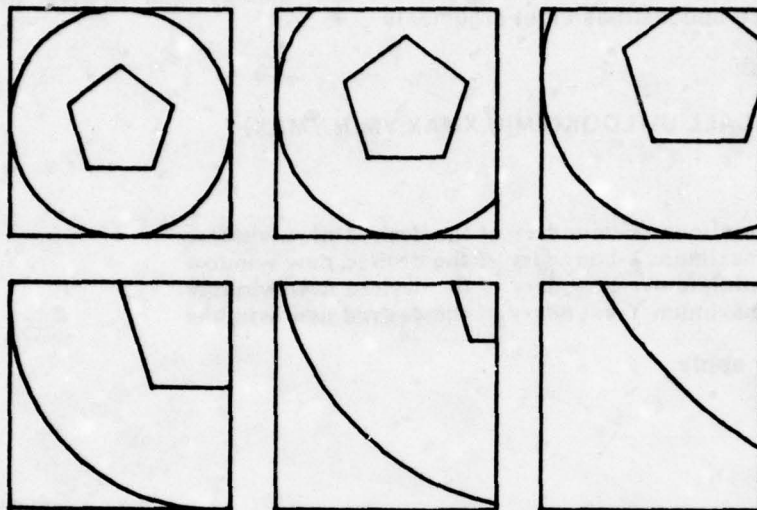
The concept of windowing into a virtual space expands greatly the capabilities of the user. It is this ability which allows him to work in his own units, plotting in a space which theoretically exists from minus infinity (largest negative number of the computer) to plus infinity (largest positive number of the computer) in both the X and Y directions, but observing only that subset of the vast space that is of interest to him. The user defines the rectangular area of interest by calling this routine (UWINDO). The user can also specify a corresponding rectangle on the physical plotting surface where any output visible in the virtual user window will appear. This area is called the user display area and may be specified by calling UDAREA. The output in virtual space is mapped on-to-one onto the user display area with resulting distortion should the ratio of the window to the display area in the X direction not be equal to that of the Y direction. Multiple user windows may be superimposed on one user display area by changing the user window without changing the user display area. Conversely multiple displays may be created from one area of virtual space by changing the user display area without changing the user window.

Subroutine UWINDO is used to set the user window boundaries at the viewplane. An error condition will occur (see Error Appendix) and the previous setting will not be modified if the maximum boundary is specified less than the minimum boundary.

In a two dimension environment, the viewplane is set to be the default Z value.

### **Programming Notes:**





```

CALL USTART
DO 1 I = 1, 6
CALL UERASE
BOUNDS = 100. - (15. * FLOAT(I-1))
CALL UWINDO (0., BOUNDS, 0., BOUNDS)
CALL UOUTLN
CALL DRWFIG
1 CONTINUE
CALL UEND
STOP
END
SUBROUTINE DRWFIG
CALL UCIRCLE (50., 50., 50.)
CALL UPLYGN (50., 50., 5., 25.)
CALL UDELL
CALL UPAUSE
RETURN
END

```

**FUNCTION:**

This routine adjusts both the virtual window and the user display area to cover the portion of virtual space specified as input arguments.

**CALLING SEQUENCE:**

**CALL UWLOOK(XMIN,XMAX,YMIN,YMAX)**

**Where**

<b>XMIN</b>	is the minimum X-boundary of the desired new window
<b>XMAX</b>	is the maximum X-boundary of the desired new window
<b>YMIN</b>	is the minimum Y-boundary of the desired new window
<b>YMAX</b>	is the maximum Y-boundary of the desired new window

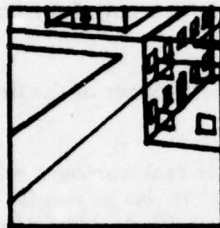
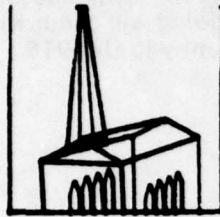
**OPTIONS which may apply:**

No options apply.

**COMMENTS:**

Should the required modification require extending the user display area beyond the display surface boundary, an error condition will be indicated and no modification will take place.

**Programming Notes:**



```

CALL ATTACH (2, '/TEXSDEND/SAVE', 'S, 8, 16T, )
CALL USTART
CALL UPSET ('TERMINATOR', '<')
CALL UPSET ('LIBRARY', 1.)
CALL UTILITY ('LOAD', 2.)
CALL UWINDO (-100., 0., 0., 100.)
CALL UDAREA (3.5, 0.5, 4.5, 7.5)
CALL UVIEW (100., -250., 00., 0., 100., 0.)
CALL UHLOOK (-100., -10., 10., 100.)
CALL UOUTLN
CALL VILLAG
CALL UHLOOK (-100., -10., -100., -10.)
CALL UOUTLN
CALL VILLAG
CALL UEND
STOP
END
SUBROUTINE VILLAG
CALL USET ('XYZ ')
CALL USET ('SYSTEMAXIS')
CALL USET ('REFERENCEAXIS')
CALL USCALL (-50., 20., 0., 1., 1., 1., 00., 0., 0., 'CHURCH')
CALL USCALL (24., -10., 0., 1., 1., 1., 00., -00., 0., 'SCHOOL')
CALL USCALL (70., 70., 0., 0.7, 0.7, 0.7, 0., 0., 0., 'PIZZA')
CALL USCALL (0., 0., 0., 1., 1., 1., 0., 0., 0., 'ROAD ')
RETURN
END

```



## **Subroutine UWRITE**

### **FUNCTION:**

This subroutine enables the user to print information at the pen position specified. The five (5) options available to the user are: 'TEXT', 'REALNUMBER', 'INTEGERNUMBER', 'XYZCOORDINATES', and 'XYCOORDINATES'. The output characters will be either hardware or software depending on the current setting in the Graphics Status Area. Margining will occur with hardware characters and windowing will occur with software characters. The beam will be restored to its position on entry to UWRITE.

### **CALLING SEQUENCE:**

**CALL UWRITE (X,Y,DATA)**

Where

- X** is the X- or RADIUS coordinate of the lower left corner of the first character of the output.
- Y** is the Y- or THETA coordinate of the lower left corner of the first character of the output.
- DATA** is a single variable or array containing either real numbers or a GCS text string. The size of DATA is variable; it is a single variable if 'REALNUMBER' or 'INTEGERNUMBER' is specified; it is a two word array if 'XYCOORDINATES' is specified; and it may be any length if 'TEXT' is specified.

### **OPTIONS which may apply:**

Alphanumeric Output Options: 'REALNUMBER', 'INTEGERNUMBER',  
'XYCOORDINATES', 'TEXT', 'XYZCOORDINATES'

Numeric Precision Option: UPSET option 'PRECISION'

Character Type Options: 'HARDWARE' 'SOFTWARE'

Character Size Options:

- for 'HARDWARE' - 'SMALL', 'MEDIUM', 'LARGE', 'EXTRALARGE'
- for 'SOFTWARE' - UPSET options 'HORIZONTAL', 'VERTICAL'

Alternate Character Format: 'GOTHIC', 'ITALIC'

Margin Boundaries: (see UMARGN)

Window Boundaries: (see UWINDO)

Coordinate System Options (see UPEN and UCOSYS)

### **COMMENTS:**

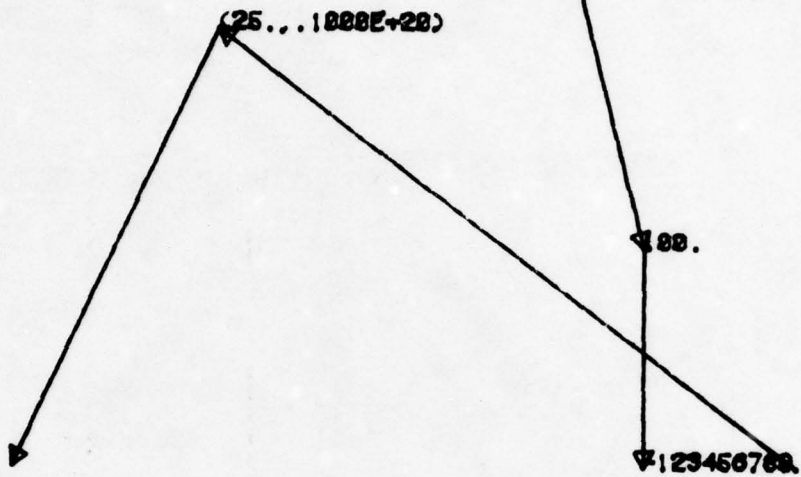
UWRITE is functionally equivalent to UPRINT except that the beam is restored to the position it occupied upon entry to UWRITE. For a complete description of all the options available using either UWRITE or UPRINT, see UPRINT.



*In three dimensional applications, the text is positioned at the point specified by X,Y,ZVALUE, where ZVALUE is established by a call to UPSET with the 'ZVALUE' option. For two dimensional applications, this corresponds to output being placed on the plane of the graphics display device.*

**Programming Notes:**

THIS IS A SAMPLE LINE OF OUTPUT TEXT



```
DIMENSION COORD(2)
DATA COORD/25.,8.88888888E 18/
CALL USTART
CALL UPSET ('TERMINATOR','')
CALL UPSET ('SPEED',128.)
CALL UERASE
CALL USET ('EXTRALARGE')
CALL UPRI NT (8.,98., 'THIS IS A SAMPLE LINE OF OUTPUT TEXT,')
CALL USET ('REALNUMBER')
CALL USET ('ARROW')
CALL UPEN (75.,58.)
CALL UWRITE (75.,58.,188.)
CALL USET ('INTEGER')
CALL UPEN (75.,25.)
CALL UPRI NT (75.,25.,-123456789.)
CALL USET ('XYCOORDINATE')
CALL UPEN (25.,75.)
CALL UWRITE (25.,75.,COORD)
CALL UPEN (1.,25.)
CALL UEND
STOP
END
```

### **Subroutine UWRIT1**

#### **FUNCTION:**

This routine displays the data provided at the current beam position in the format currently specified in either hardware or software characters subject to the one-time setting of the USET option furnished which only applies during the execution of this subroutine. Upon return the beam is positioned at the location it held upon entering this subroutine.

#### **CALLING SEQUENCE:**

**CALL UWRIT1 (DATA,OPTION)**

Where

**DATA** is a single variable or array containing either real numbers or a GCS text string. The size of DATA is variable: it is a single variable if 'REALNUMBER' or 'INTEGERNUMBER' is specified; it is a two-word array if 'XYCOORDINATES' is specified it is a three word array if 'XYZCOORDINATES' is specified; and it may be any length if 'TEXT' is specified.

**OPTION** is a Hollerith string variable or literal defining the USET option to apply during the execution of this subroutine.

#### **OPTIONS which may apply:**

Same as those for UWRITE except that coordinate system options do not apply.

#### **COMMENTS:**

Calling UWRIT1 is equivalent to the sequence of calling USET with OPTION, calling UWRITE at the current beam position with DATA, and then calling USET with the appropriate options which would restore the status changed by OPTION. UWRIT1 is convenient for producing alphanumeric information at points being plotted:

#### **Programming Notes:**

```

CALL USTART
CALL UPSET ('TERMINATOR',',','')
CALL UASPCT (1.)
CALL USET ('SOFTWARE CHARACTERS')
CALL UPSET ('HORIZONTAL SIZE',2.5)
CALL UPSET ('VERTICAL SIZE',3.5)
CALL UMOVE (50.,50.)
DO 10 I = 1, 8
CALL UROTAT (45.*(FLOAT(I)-1))
10 CALL UWRIT (' CIRCULAR MESSAGE',',','TEXT')
CALL UEND
STOP
END

```



*Subroutine UZWDO*

**FUNCTION:**

This routine specifies the hither and yon boundary planes of the user window without altering the currently specified X and Y window boundaries.

**CALLING SEQUENCE:**

**CALL UZWDO(ZMIN,ZMAX)**

Where

<b>ZMIN</b>	is the hither or minimum Z-boundary of the new window.
<b>ZMAX</b>	is the yon or maximum Z-boundary of the new window.

**OPTIONS which may apply:**

Z axis clipping: 'NOZCLIPPING', 'ZCLIPPING'

**COMMENTS:**

A new window boundary specification is set in which the new X and Y boundaries are the same as the old X and Y boundaries and the new Z boundaries are obtained from the input parameters. If ZMAX .LT. ZMIN, an error is generated and no change is made in the window. Clipping on the Z boundaries will only occur if 'ZCLIPPING' is specified (see U3WDO).

**Programming Notes:**

**FUNCTION:**

This routine specifies the screen location within which the contents of the user window (U3WNDO) will be displayed.

**CALLING SEQUENCE:**

**CALL U3AREA(XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)**

Where

<b>XMIN</b>	is the minimum X boundary in current device units of the new user display area
<b>XMAX</b>	is the maximum X boundary in current device units of the new user display area
<b>YMIN</b>	is the minimum Y boundary in current device units of the new user display area
<b>YMAX</b>	is the maximum Y boundary in current device units of the new user display area
<b>ZMIN</b>	is the minimum Z boundary in current device units of the new user display area
<b>ZMAX</b>	is the minimum Z boundary in current device units of the new user display area

**OPTIONS which may apply:**

Device Units: 'INCHES', 'CENTIMETERS', 'PERCENT UNITS', 'RASTER', 'FONT',  
'SPECIFICATIONS'

**COMMENTS:**

This routine works in close conjunction with U3WNDO to define the mapping to take place between virtual space and device space. UDAREA defines a rectangular paralleliped which is a subset of the total device space. The six-sided viewing solid defined by U3WNDO and the current projection mode (perspective or orthogonal) is mapped into this parallelopiped.

The units used to describe the desired region of the display area are device, rectangular, absolute units. Any of the device units may be used (e.g., 'INCHES', 'CENTIMETERS', etc.). If any of the maximum boundary parameters specifies a boundary less than or equal to the minimum boundary parameters, an error is generated and the old U3AREA specification is retained.

If a display-device with only a 2-D address space is being utilized, the Z values are checked for validity but are not used. The projection of the U3WNDO onto the view plane will then be mapped into the U3AREA X and Y space of the screen plane. The U3AREA is frequently referred to as the 'viewport' in the literature.

**Programming Notes:**

**FUNCTION:**

This routine creates a set of coordinate axes in 3-space.

**CALLING SEQUENCE:**

CALL U3AXIS(XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)

Where

XMIN	is the minimum value of the X axis
XMAX	is the maximum value of the X axis
YMIN	is the minimum value of the Y axis
YMAX	is the maximum value of the Y axis
ZMIN	is the minimum value of the Z axis
ZMAX	is the maximum value of the Z axis

**OPTIONS which may apply:**

Origin Inclusion Options: 'ORIGIN', 'NOORIGIN', 'PENORIGIN'

View Adjustment Options: 'XYVIEW', 'XZVIEW', 'YZVIEW', 'XYZVIEW'

Grid Type Specifications: UPSET('GRID',value)

Axis Existence Options: 'NOAXES', 'XAXIS', 'YAXIS', 'ZAXIS', 'XYAXES', 'XZAXES',  
'YAXES', 'XYZAXES'

Axis Scaling Options: 'AUTOSCALE', 'FULLSCALE', 'OWNSCALE'

Coordinate Type Options: 'RECTANGULAR', 'POLAR', 'CYLINDRICAL', 'SPHERICAL'

Logarithmic Transforms: 'NOLOG', 'XLOG', 'YLOG', 'ZLOG', 'XYLOG', 'XZLOG',  
'YZLOG', 'LOGARITHMIC'

Logarithmic Application Time: 'LOGSYSTEM', 'LOGUSER', 'LOGORIGINAL'

Logarithm Base: UPSET 'BASE', 'XBASE', 'YBASE', 'ZBASE'

**COMMENTS:**

This routine is designed to supercede the 2-D URAXIS routine.

U3AXIS performs all operations in virtual space. This means that all axes and labels will be clipped if they are outside of the viewing pyramid. Which viewing pyramid will be used will depend upon the View Adjustment Options. If 'XYZVIEW' is specified, the current view will be used. This is the default. If 'XYVIEW', 'XZVIEW', or 'YZVIEW' is specified, a viewing vector is defined which is at positive 150. along the axis perpendicular to the designated plane. The view site is defined to be at (0,0,0) in current units. Note that UWINDO/UDAREA modifications will only occur if 'XYZVIEW' is not specified.

Scaling options allow the user to select the intervals or allow U3AXIS to choose them. The Origin Inclusion Option allows the user to deselect the forced origin of the 'AUTOSCALE' and 'FULLSCALE' options. The default, 'ORIGIN', forces an origin for these options. 'NOORIGIN' will now allow the natural range of numbers of the U3AXIS

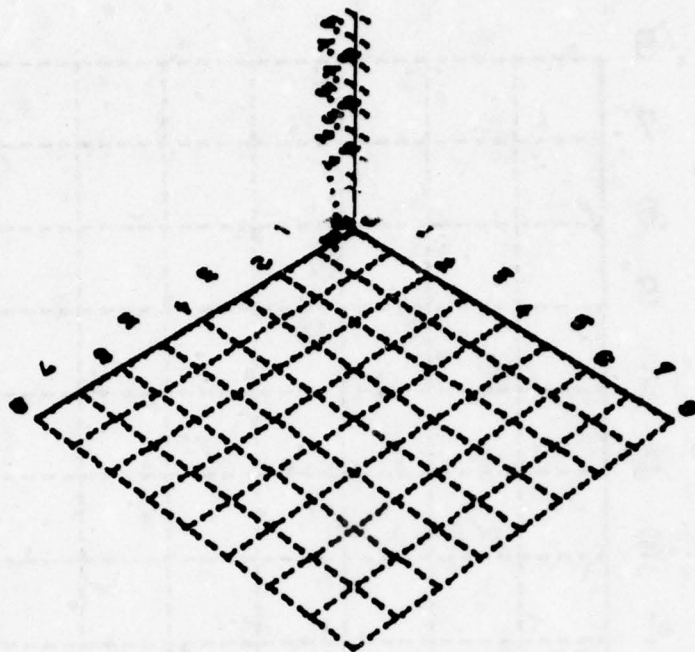


input arguments to determine the axis range. Note that under the 'OWNSCALE' option, origins are never forced. Use of the 'PENORIGIN' option allows the user to force the current pen position to be included in the axis range.

Grids will only be generated in the view adjustment plane (if specified) or the X-Y plane if 'XYZVIEW' is specified. The type of line to be used can be user selected by calling UPSET('GRID',value) where value is either 0. or a valid dash specification. If the default value of 0. is specified, a solid line will be used to create the grids.

**Programming Notes:**

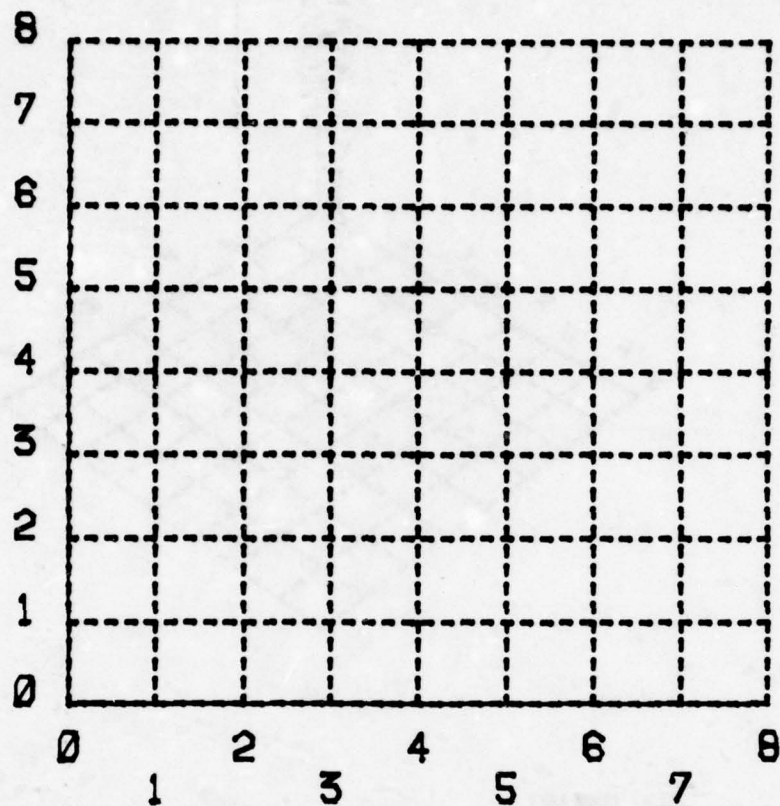




```

CALL USTART
CALL UWINDO (-8.,8.,-8.,8.)
CALL UVIEW (30.,30.,30.,2.,2.,0.)
CALL UPSET ('GRID SPECIFICATION',34.)
CALL USET ('GRIDAXES')
CALL USET ('SOFTWARE CHARACTERS')
CALL UPSET ('HORIZONTAL',.3)
CALL UPSET ('VERTICAL',.5)
CALL UPSET ('LABEL',90.)
CALL UPSET ('TICPLUS',.3)
CALL UPSET ('TICMINUS',.3)
CALL USAXIS (0.,8.,0.,8.,0.,5.)
CALL UEND
STOP
END

```



```

CALL USTART
CALL UWINDO (-8.,8.,-8.,8.)
CALL UVIEW (30.,30.,30.,2.,2.,8.)
CALL UPSET ('GRID SPECIFICATION',34.)
CALL USET ('GRIDAXES')
CALL USET ('SOFTWARE CHARACTERS')
CALL UPSET ('HORIZONTAL',.3)
CALL UPSET ('VERTICAL',.5)
CALL UPSET ('LABEL',90.)
CALL USET ('XYVIEW')
CALL U3AXIS (0.,8.,0.,8.,0.,5.)
CALL UEND
STOP
END

```

**FUNCTION:**

This routine invokes an already constructed graphics data structure. The requested coordinate system transformation is established, then each element of the structure is executed.

**CALLING SEQUENCE:**

**CALL U3CALL(X,Y,Z,SX,SY,SZ,RX,RY,RZ,NAME)**

Where

**X,Y,Z** are the coordinates of the origin of the structure coordinate system in current units.

**SX,SY,SZ** are the scale factors along the axes respectively.

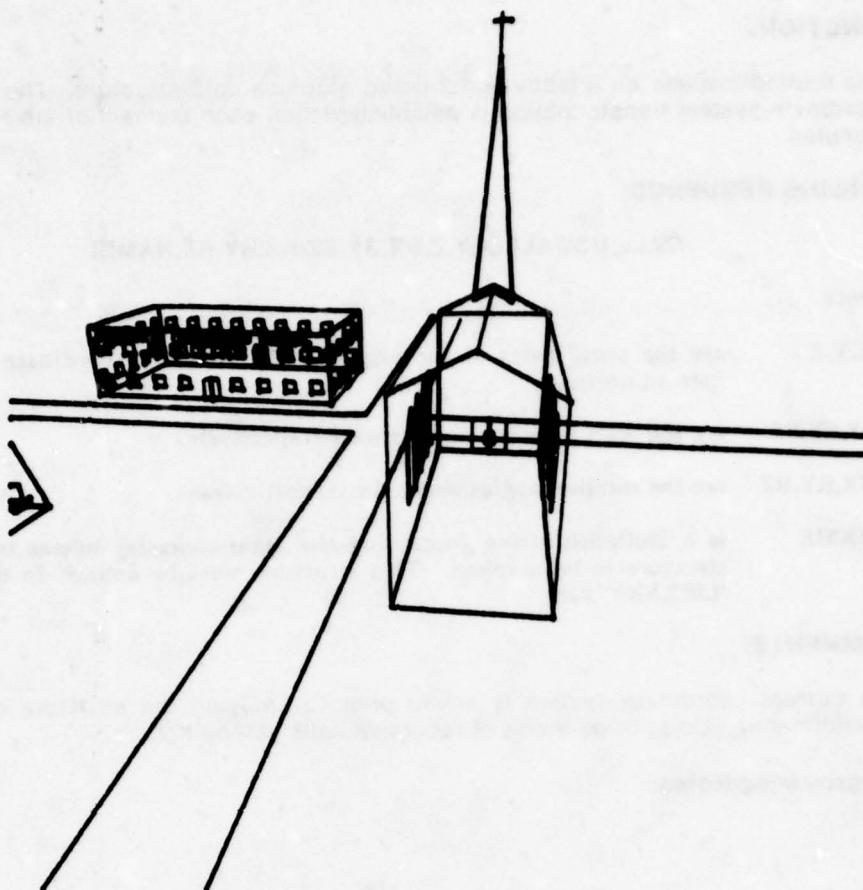
**RX,RY,RZ** are the rotation angles about the respective axes.

**NAME** is a Hollerith string containing the eight-character names of the data structure to be invoked. This structure must be defined in the current 'LIBRARY' file.

**COMMENTS:**

The current coordinate system is saved prior to applying the structure coordinate transformation. Up to three levels of recursive calls can be handled.

**Programming Notes:**



```

CALL ATTACH (8, '/TEK3DEMO/SAVE', '3,0,IST,')
CALL USTART
CALL UPSET ('LIBRARY', 1.)
CALL UTILITY ('LOAD', 8.)
CALL UPSET ('TERMINATOR', '<')
CALL USET ('VIEWDISTANCE')
CALL UVMPRT (150.)
CALL UWINDO (-100., 100., -100., 100.)
CALL UVIEW (-40., 200., 70., -20., 20., 0.)
CALL VILLAG
CALL UEND
STOP
END
SUBROUTINE VILLAG
CALL USET ('XYZ')
CALL USET ('SYSTEMAXIS')
CALL USET ('REFERENCEAXIS')
CALL USET ('BLACK')
CALL U3CALL (-50., 20., 0., 1., 1., 1., 90., 0., 0., 'CHURCH')
CALL USET ('RED')
CALL U3CALL (24., -10., 0., 1., 1., 1., 90., -90., 0., 'SCHOOL')
CALL USET ('BLUE')
CALL USET ('GREEN')
CALL USET ('YELLOW')
CALL USET ('MAGENTA')
CALL USET ('CYAN')
CALL USET ('BLACK')
END

```



**FUNCTION:**

This routine composes a new user coordinate system based on the current reference coordinate system using the input arguments for translation, scaling, and rotation operations.

**CALLING SEQUENCE:**

**CALL U3CSYS(X,Y,Z,SX,SY,SZ,RX,RY,RZ)**

Where

**X,Y,Z** the coordinates of the new origin in units of the old coordinate system.

**SX,SY,SZ** are the multiplicative scale factors along each of the axes.

**RX,RY,RZ** indicate the amount of rotation around each of the axes in current angular units.

**OPTIONS:**

(Only those which differ from SUBROUTINE UCOSYS are listed)

Rotation Application Order: 'XYZ', 'XZY', 'ZXY', 'ZYX', 'YXZ', 'YZX'.

**COMMENTS:**

The composition of user coordinate systems is a powerful tool for use in describing problem pictures in their own environment. A full description of the use of this facility is available under UCOSYS and a tutorial is available in the primer. In using this facility in three-dimensions, only one additional requirement is imposed on the user: specification of the order in which the rotations will occur. This is a result of the non-commutativeness of the rotation operation. To set the order, the user merely specifies, as a USET option, the three characters which designate the axes in the order in which he wishes the rotations to be applied. The default setting is CALL USET('ZYX').

The positive direction of rotation is counterclockwise around the axis of rotation when viewed from a position on the axis of rotation looking in the negative direction.

**Programming Notes:**

**FUNCTION:**

This routine draws a solid line vector in three-space from the current pen position to the position specified by the input arguments.

**CALLING SEQUENCE:**

**CALL U3DRAW(X,Y,Z)**

Where

**X,Y,Z** are the coordinates of the end point of the line in current units.

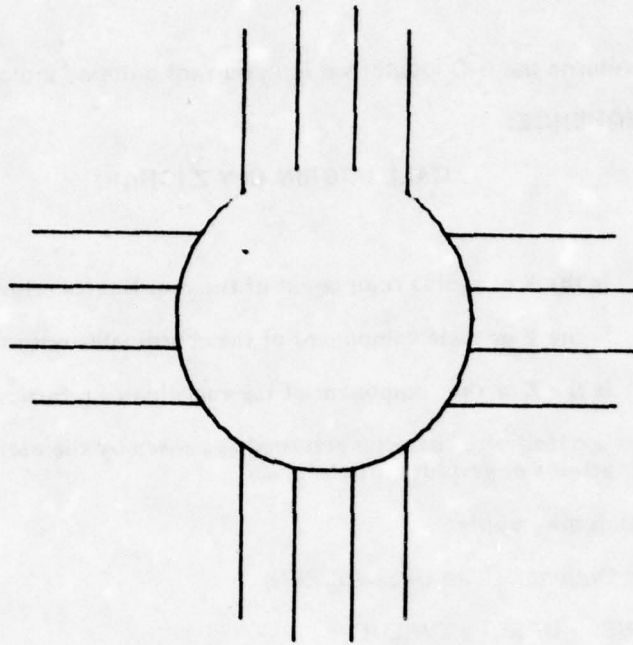
**OPTIONS:**

Pen Coordinate Options (see U3PEN)

**COMMENTS:**

This routine is equivalent to using U3PEN with line option 'LNULL'. The current setting of the line option has no effect on this routine.

**Programming Notes:**



ZIA Sun Symbol

```

DIMENSION X(16),Y(16)
DATA X/20.,20.,0.,0.,0.,0.,-20.,-20.,-20.,-20.,0.,0.,0.,0.,
&      20.,20./
DATA Y/0.,0.,20.,20.,20.,20.,0.,0.,0.,0.,-20.,-20.,-20.,-20.,
&      0.,0./
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UCOSYS (50.,50.,1.,1.,0.)
CALL USET ('POLAR')
CALL UMOVE (17.5,0.)
DO 10 I = 1, 120
10 CALL UDRAW (17.5,FLOAT(I)*3.)
  THETA = 11.5
  DO 20 I = 1, 16
    CALL USET ('POLAR')
    CALL USET ('ABSOLUTE')
    CALL UMOVE (17.5,THETA)
    CALL USET ('RECTANGULAR')
    CALL USET ('RELATIVE')
    CALL UDRAW (X(I),Y(I))
  20 THETA = 11.25 + 22.5 * FLOAT(I)
    CALL USET ('LARGE')
    CALL USET ('JUSTIFICATION')
    CALL USET ('ABSOLUTE')
    CALL UPRINT (0.,-45., 'ZIA S>UN <S>YMBOL,')
    CALL UEND
  STOP
  END

```

### **Subroutine U3GRIN**

#### **FUNCTION:**

This routine returns the 3-D locator value in current units as indicated by the user.

#### **CALLING SEQUENCE:**

**CALL U3GRIN (X,Y,Z,ICHAR)**

Where

<b>X</b>	is the X or radius component of the coordinates returned
<b>Y</b>	is the Y or theta component of the coordinates returned
<b>Z</b>	is the Z or rho component of the coordinates returned
<b>ICHAR</b>	is a Hollerith character returned as chosen by the user or derived from the actions of graphics input device

#### **OPTIONS which may apply:**

All coordinate Options: (see UPEN/U3PEN)

Default ZVALUE: 'UPSET', 'ZVALUE'

Graphics INput Device: 'CURSOR', 'JOYSTICK', 'LIGHTPEN', 'TABLET', 'KEYBOARD',  
'FUNCTION KEY', 'MOUSE', 'TRACKBALL'

#### **COMMENTS:**

If only a 2-D locator is available, the third component will be obtained from the current Z-VALUE setting.



**FUNCTION:**

This routine applies a general three-dimensional image transformation to the indicated segment/frame.

**CALLING SEQUENCE:**

**CALL U3IMAG (X,Y,Z,SX,SY,SZ,RX,RY,RZ,SEGID)**

**Where**

**X,Y,Z** is the new position of the segment in current 3D device units.

**SX,SY,SZ** is the scale factor to be applied along each axis of the display surface.

**RX,RY,RZ** in the rotation in current angular units to be applied around each axis of the display surface.

**SEGID** is the identifier of a "retained" segment/frame which was UOPENed for general 3D image transformation.

**OPTIONS which may apply:**

Device Units: 'INCHES', 'CENTIMETERS', 'RASTERUNITS', 'FONTUNITS',  
'SPECIFICATION UNITS', 'PERCENT UNITS'

Specification Unit Size (UPSET): 'SPECIFICATION UNITS', 'XSPECIFICATION UNITS',  
'YSPECIFICATION UNITS', 'ZSPECIFICATION UNITS'

Angular Units: 'DEGREES', 'RADIAN', 'PIRADIAN', 'GRADS', 'MILS'

Segment Identifier Mode: 'FNAME', 'FNUMBER', 'SNAME', 'SNUMBER'

**COMMENTS:**

The image transformation is applied in the order X-rotation, Y-rotation, Z-rotation, scaling, and translation. If the resultant image should exceed the display surface address space, the result is undetermined.

Image transformations are only applied if supported on the current display surface. Requests for image transformation will be ignored if the display device does not support this facility.

**Programming Notes:**

**FUNCTION:**

This routine creates a line in 3-space by connecting an array of coordinates with lines drawn with the current line option.

**CALLING SEQUENCE:**

**CALL U3LINE(X,Y,Z,PTS)**

Where

- X** is an array of length PTS containing the X-components of the coordinates of the points to be connected.
- Y** is an array of length PTS containing the Y-components of the coordinates of the points to be connected.
- Z** is an array of length PTS containing the Z-components of the coordinates of the points to be connected.
- PTS** is a real variable which specifies the number of points to be connected.

**OPTIONS which may apply:**

All pen-related options (see U3PEN)

**COMMENTS:**

This routine will move to the first point specified and then draw a line to succeeding points until the last point is reached. The pen or beam will be left at the last point.

**Programming Notes:**

**FUNCTION:**

This routine draws an invisible vector in three-space from the current pen position to the position specified by the input arguments.

**CALLING SEQUENCE:**

**CALL U3MOVE(X,Y,Z)**

Where

**X,Y,Z** are the coordinates of the end point of the line in current units.

**OPTIONS:**

Pen Coordinate Options (see U3PEN)

**COMMENTS:**

This routine is equivalent to using U3PEN with line option 'NNULL'. The current setting of line option has no effect on this routine.

**Programming Notes:**

**FUNCTION:**

This routine draws a 3D vector of the type indicated by the current line option from the current beam position (vector tail) to the location specified by the arguments (vector head). The current beam position is updated to point to the head of the vector.

**CALLING SEQUENCE:**

**CALL U3PEN (X,Y,Z)**

Where

- X** is the X or RADIUS coordinate component of the head of the vector in current user units.
- Y** is the Y or THETA coordinate component of the head of the vector in current user units.
- Z** is the Z or RHO coordinate component of the head of the vector in current user units.

**OPTIONS which may apply:**

Line Definition Options

Line Options

Tic Interval: UPSET('TICINTERVAL',value)

Tic Mark Size Options: UPSET('TICPLUS',value) UPSET('TICMINUS',value)

Dash Specification: UPSET('SETDASH',value)

System Character Setting: UPSET('MARKER',value)

Character Description Options: see U3PRNT.

Symbol/Marker Selection: UPSET('MARKER',value) UPSET('SYMBOL',value)

Symbol/Marker Size: UPSET('SZMARKER',value)

Coordinate Output Type: '2DCOORDINATES', '3DCOORDINATES'

Gapped Line Mode: 'GAPPED', 'UNINTERRUPTED'

Mapping/Projection Options: 'PERSPECTIVE', 'ORTHOGRAPHIC'

Pen Coordinate Options

Type: 'RECTANGULAR', 'POLAR', 'CYLINDRICAL', 'SPHERICAL'

Mode: 'ABSOLUTE', 'RELATIVE', 'XRELATIVE', 'YRELATIVE', 'ZRELATIVE',  
'XABSOLUTE', 'YABSOLUTE', 'ZABSOLUTE', 'XYRELATIVE',  
'XZRELATIVE', 'YZRELATIVE', 'XYABSOLUTE', 'XZABSOLUTE',  
'YZABSOLUTE'



Space: 'VIRTUAL', 'DEVICE'

Device Space Units: 'PERCENT', 'INCHES', 'CENTIMETERS', 'RASTER',  
'FONT', 'SPECIFICATION'

Specification Unit Size: UPSET('SPECIFICATION',value)  
UPSET('XSPECIFICATION',value)  
UPSET('YSPECIFICATION',value)  
UPSET('ZSPECIFICATION',value)

Coordinate System Selection: 'SYSTEM', 'WORLD', 'USER', 'MODELLING'

User Coordinate System Type: 'REFERENCE', 'WORKING'

Logarithmic Transforms: 'LOGARITHMIC', 'NOLOGARITHMS',  
'XLOGARITHMIC', 'YLOGARITHMIC',  
'ZLOGARITHMIC', 'XYLOGARITHMIC',  
'XZLOGARITHMIC', 'YZLOGARITHMIC',  
'XYZLOGARITHMIC'

Logarithm Application Time: 'LOGOBJECT', 'LOGUSER', 'LOGSYSTEM'

Logarithm Base: UPSET('BASE',value), UPSET('XBASE',value),  
UPSET('YBASE',value), UPSET('ZBASE',value)

**ATTRIBUTES:**

Intensity 'BRIGHT', 'DIM', 'NORMAL',  
UPSET('BRIGHTNESS',value)

Line Width 'THIN', 'WIDE', UPSET('WIDTH',value)

#### **COMMENTS:**

U3PEN is the central pen/beam movement subroutine of the three dimensional version of GCS. Not only is it called frequently by users in creating graphics images, it is also used extensively within the higher-level GCS routines.

To provide its power and flexibility, a large number of USET and UPSET options may effect any particular call to U3PEN. However, this should not be a concern for the less-experienced user since the defaults for these options will produce the effect he most likely desires, thus relieving him of the requirement to set any options but those which apply to this specific program.

These options have been classified into several major categories, each of which will be explained in detail. The visible output from U3PEN will also be effected by the viewing environment if plotting is occurring in virtual space. A full description of the concepts of the viewing environment can be found under UVIEW and UWINDO.

#### **Programming Notes:**

## LINE DEFINITION OPTIONS:

This options apply to only the physical aspects of the lines created by U3PEN. Attributes which apply to both lines and characters are described under Attributes.

## LINE OPTIONS:

The 40 available line options can be easily remembered if each is divided into its two component parts, the line type and the terminator type. The following table lists the available line and terminator types. Any line type may be combined with any terminator type to create a particular line option. This is done by prefixing the name of the terminator with the first character of the name of the line type. For example, to create a dashed arrow the first character of the line type "dashed" is "D". This letter is affixed to the word "arrow" (the name of the terminator) and the resulting USET line option is "DARROW". A ticked arrow would result if a "T" was substituted for the "D" to yield "TARROW". Calling UPEN with line option "NNULL" will yield identical results to calling UMOVE. Examples of the different line options are shown in Figure 1.

Line Type	Terminator Type
Null	NULL (absence of a terminator)
Line	POINT
Dashed	ARROW
Ticked	BACKARROW
Alpha	CHARACTER
	SYMBOL
	COORDINATES

Also available for the convenience of the user are synonyms for the most frequently selected line option. A table of these synonyms with their corresponding line/terminator specifications is included below:

Synonym	Line/Terminator Equivalent
ALPHALINE	ANULL
ARROW	LARROW
BACKARROW	LBACKARROW
CHARACTER	NCHARACTER
DASH	DNULL
DIMENSIONLINE	LDOUBLEARROW
DOUBLEARROW	LDOUBLEARROW
LINE	LNULL
MOVE	NNULL
NO LINE	NNULL
NO MARK	NNULL
NOMARK	NNULL
NOLINE	NNULL
PENDOWN	LNULL
PENUP	NNULL
POINT	NPOINT
SYMBOL	NSYMBOL
TICLINE	TNULL

Several other options may effect the appearance of some line options. These are tic-interval settings, dash specifications and system character settings. Each of these is described in more detail below.

## TIC-INTERVAL SETTINGS:

The tic interval to be used when drawing ticked lines can be set by the UPSET option

"TICINTERVAL". The UPSET parameter value should be the distance between tic values in current user units (see Pen Coordinate Options below.) The user is cautioned that if he changes his space from device to virtual or visa versa then this may invalidate his tic-interval setting. Figure 2 shows the use of the "TICINTERVAL" UPSET parameter.

#### **TIC MARK SIZE OPTIONS:**

The size of the individual tic marks may be specified. GCS allows the tic marks to be of different sizes on each side of the line. If the line along which the tic marks are drawn is considered to be an X-axis and is rotated about its tail in the XY plane. When viewed from the positive Z-direction, the side towards positive angles of rotation is considered the 'TICPLUS' side and the side towards negative angles of rotation is considered the 'TICMINUS' side. For 2D applications, the 'TICPLUS' side is in the positive Y direction and the 'TICMINUS' side is towards the negative Y direction if the line is considered to be an X-axis. The tic mark sizes may then be set by the following UPSET options:

UPSET('TICPLUS',value) — The 'TICPLUS' size is set to the length value specified in current units.

UPSET('TICMINUS',value) — The 'TICMINUS' size is set to the length value specified in current units.

#### **DASH SPECIFICATION:**

The dash line specification which UPEN will use can be adjusted to produce a wide variety of dashed lines. This adjustment is made by the UPSET option 'SETDASH' where the parameter value is a number which defines the dashed line as follows. Single digits allow selection of hardware-generated dashed lines if that capability exists. The digits 1-8 will produce up to 8 different dashed lines depending on the capability of the hardware. The digit 9 will produce a hardware-dotted line. Should hardware dashed lines not be available on the terminal being used, a default software dashed line will be generated.

Software dashed line specifications consist a floating print number, made of two or more digits which are evaluated left to right, digit by digit. Each digit represents a line segment which is either visible or invisible of varying length as specified in the following table.

Digit	Length (inches)	Visibility
1	.0366	Visible
2	.0366	Invisible
3	.0733	Visible
4	.0733	Invisible
5	.1831	Visible
6	.1831	Invisible
7	.3662	Visible
8	.3662	Invisible
9	.0073 (dot)	Visible

When the entire integer has been evaluated, scanning restarts at the left most digit. For example, dash code 34. will result in a visible line of .0733 inches followed by an invisible line of .0733 inches followed by a visible line of .0733 inches, etc., continuing until the head of the vector is reached. The last segment may be a shortened visible or invisible segment. If, however, the next UPEN movement is a dashed line starting at the head of the previous vector, the last segment of this previous vector will be completed and the sequence continued for the new vector.

The complexity of the dashed line specification is limited only by the number of digits of precision available in the computer floating point numbers. Moreover, visible line



segments need not be alternated with invisible line segments. Visible line segments may be combined with other visible line segments to create line segments of length not available through single digits. Similarly, this applies to invisible line segments.

#### **SYSTEM CHARACTER SETTING:**

The system character is used to create ALPHA lines and to indicate which character should be used to terminate UPEN operations when the line option specifies the 'CHARACTER' terminator. The system character may be set by a call to USET where the argument is the single character to become the system character. For example, the following statement would set the system character to the dollar sign:

CALL UPSET ('CHARACTER','\$')

Should it also be desired to change the line type as well as the system character, the character to be specified in the USET argument may be prefixed with one of the five available line types. The following statement would change the system character to "+", and would also change the line option to 'DCHARACTER':

CALL USET ('D+')

The default system character is the asterisk (\*).

Also to be considered is the case in which GCS is set at the time the system character is set. The default case is UPPERCASE. However, if calls to UPRINT have left GCS in LOWERCASE mode, any attempt to set the system character will generate a lower case character. The desired case may be set by using one of the following

CALL USET('UPPERCASE')  
CALL USET('LOWERCASE')

See UPRINT for a complete description of case shifting.

#### **CHARACTER TERMINATOR:**

The character terminator which can be attached to any of the line types will appear as either a hardware character, a simulated hardware character, or a software character depending on the current user setting of 'HARDWARE', 'SIMULATED' or 'SOFTWARE'. Hardware characters and simulated hardware characters will have their lower-left corner on the head of the vector. Software characters will have the head of the vector at the center of the character. The appearance of hardware, simulated hardware, or software characters may be affected by the setting of the italicization and character font option if the feature is available and has been implemented. Regardless to which type of character is specified, at the end of the UPEN operation, the beam on pen will be located at the head of the vector. See U3PRNT for a more detailed description of the character options.

#### **CHARACTER DESCRIPTION OPTIONS:**

The character description options are fully described in U3PRNT. Most character description options apply when using character or coordinate line terminators.

#### **SYMBOL/MARKER TERMINATOR:**

The symbol terminator is available so that the location of the head of each line may be uniquely marked. There are two types of symbols. The first type are GCS-defined symbols. The second type are installation-defined or device-defined symbols. The symbol is always displayed centered at the head of the line and the resulting beam/pen position is at the head of the line. Should a marker protrude over a display surface edge



or the edge of a window, the results are undefined (i.e., the marker may be partially slipped or may wraparound).

#### **SYMBOL/MARKER SELECTION:**

Symbols are selected by `UPSET('MARKER',index)` or `UPSET('SYMBOL',index)` where *index* is a non-negative real number whose integer value is used to select the symbol. Specifications of an index outside of the range of valid indices will produce symbol X.

#### **SYMBOL/MARKER SIZE:**

The size of symbols/markers may be specified by `UPSET('SZMARKER',value)` where *value* is a positive number in current device units. Markers generated by hardware symbol generators may be restricted as to size variations. If so, the closest size (perhaps the only size) will be used.

#### **COORDINATE TERMINATORS:**

The coordinate terminators will generate a set of coordinates equivalent to the location value of the head of the vector in current user units enclosed in parentheses and separated by a comma. The result will be in the same format as if the coordinates had been produced by a call to `UPRINT`. For a more complete discussion of this format see `UPRINT`.

#### **COORDINATE OUTPUT TYPE:**

The dimensionality of coordinate output is determined by the setting of the `USET` options '2DCOORDINATES' and '3DCOORDINATES'. If '2DCOORDINATES' are specified, then only the (X,Y) or (R,THETA) components are displayed. With '3DCOORDINATES', all three coordinate components are displayed.

#### **GAPPED LINE MODE:**

If 'GAPPED' lines are specified, successive calls to `U3PEN` will alternate drawing lines of the current line option with 'NNULL' (invisible) lines. This feature can be deactivated by specifying 'UNINTERRUPTED' lines which are the default.

#### **MAPPING/PROJECTION OPTIONS:**

The user may specify the type of mapping to be applied in projecting three-dimensional lines onto the viewport. The two available mappings are 'ORTHOGRAPHIC' in which the lines of projection are parallel and perpendicular to the projection plane, and 'PERSPECTIVE' in which line length diminishes as the distances from the viewing position (view point) becomes greater. 'PERSPECTIVE' is the default. The difference between these two options is shown in Figure 9.

#### **PEN COORDINATE OPTIONS:**

The GCS Pen Coordinate Options have been designed to provide the flexibility which will allow the user to work in his own coordinate type and unit while retaining the simplicity of operation and specification which the infrequent and/or unsophisticated user will find convenient. Thus, the Pen Coordinate Options have been divided into several classes each of which allows setting of its options completely independently of any effect on any of the others. The following paragraphs will discuss each class in detail.

#### **COORDINATE TYPE:**

This indicates the kind of coordinates the user will specify and how they will be interpreted for display.

- 'RECTANGULAR'** — Coordinates are of the standard forms (X,Y) or (X,Y,Z) where X is the number of units along the X-axis, Y is the number of units along the Y-axis, and Z is the number of units along the Z-axis. This is the standard default.
- 'POLAR' 'CYLINDRICAL'** — Coordinates are of the forms (R,THETA) or (R,THETA,Z) where R is the number of units of radius in the XY-plane, THETA is the number of angular units from the X-axis in the XY-plane, and Z is the number of units along the Z-axis. (See UARC for a description of angular units.)
- 'SPHERICAL'** — Coordinates are of the form (RHO, THETA, PHI) where RHO is the number of units of radius, THETA is the number of angular units from the X-axis around the Z-axis in the XY-plane, and PHI is the number of angular units from the Y-axis around the X-axis.

#### **COORDINATE MODE:**

This indicates where the base point or origin of the coordinate units is located. Since the conversion from relative to absolute occurs first, the (X,Y,Z) components described below may be (R,THETA,Z) or (RHO,THETA,PHI) components if the user has specified a coordinate which is not rectangular.

- 'ABSOLUTE'** — All coordinate components are indicated based on the origin of the current coordinate system. This is the default case
- 'RELATIVE'** — All coordinates are indicated with respect to the current beam/pen position. At the completion of any beam/pen movement, the beam will be at relative coordinates (0,0,0).
- 'XABSOLUTE' 'YZRELATIVE'** — The X-component is absolute and the Y- and Z-components are relative.
- 'YABSOLUTE' 'XZRELATIVE'** — The Y-component is absolute and the X- and Z-components are relative.
- 'ZABSOLUTE' 'XYRELATIVE'** — The Z-component is absolute and the X- and Y-components are relative.
- 'XRELATIVE' 'YZABSOLUTE'** — The X-component is relative and the Y- and Z-components are absolute.
- 'YRELATIVE' 'XZABSOLUTE'** — The Y-component is relative and the X- and Z-components are absolute.
- 'ZRELATIVE' 'XYABSOLUTE'** — The Z-component is relative and the X- and Y-components are relative.

**NOTE:** Use of relative coordinates will not necessarily generate device commands using relative coordinates.

#### **COORDINATE SPACE:**

This indicates the address space to which the coordinates refer.

**'VIRTUAL'**

- The coordinates indicate a point in virtual space. Whether this point is visible on the display surface will depend on the current viewing environment (for a complete discussion of the viewing environment, see UVIEW and UWINDOW). For 2-D applications, only the UWINDOW settings need be considered. These coordinates can encompass any of the real numbers within the capabilities of the computer system. No meaning is attached to these numbers by GCS; therefore, the user is free to assign whatever meaning he desires to the numbers.

**'DEVICE'**

- The coordinates indicate a point on the 'DISPLAY' physical device surface. Specification of points not on the display surface may yield indeterminate and/or spurious results. Since the lower left front corner of the display surface always has address (0,0,0), all negative coordinates may yield spurious results. The units which these numbers represent are described in the next paragraph.

**DEVICE SPACE UNITS:**

This indicates the interpretation to be given to the numbers used in plotting in 'DEVICE' space (see previous paragraph). These options have no meaning and are not considered when plotting in 'VIRTUAL' space.

**'INCHES'**

- The coordinate values refer to actual inches on the display surface. This will be true regardless of the device being used. These units are device-dependant since the size of the display surface can vary from device to device.

**'CENTIMETERS'**

- The coordinate values refer to actual centimeters on the display surface. This will be true regardless of the device being used. These units are device-dependent since the size of the display surface can vary from device to device.

**'RASTER UNITS'**

- The coordinate values refer to the individual addressable positions on any particular device. These units are device-dependant and may not be transferable to other terminal devices since the size of the device address space may vary from device to device.

**'FONTUNITS'**

- The coordinate values refer to the lower left front corner of individual character positions. The size of a character position is dependent upon the current size of the hardware, simulated hardware, or software characters (whichever is currently selected). It should be noted that these units need not be integers. On most terminals, characters can be placed anywhere on the display surface. These units are device-dependent since the size of hardware characters and the size of the display surface can vary from device to device. For a complete description of hardware, simulated hardware, and software characters, see U3PRNT.

**'PERCENTUNITS'**

- The coordinate values refer to hundredth increments of the display surface address space. This is truly a device-dependent unit. Moving an image created on one device in 'PERCENTUNITS' to another device will display an image using the same proportional amount of the display



surface. It should be noted that if a device does not have a square display surface (aspect ratio = 1), the value of a 'PERCENTUNIT' in the X-direction will differ from the value of a 'PERCENTUNIT' in the Y-direction. See UASPCT for a description of the effect on 'PERCENTUNITS' of setting the aspect ratio of the display surface.

**'SPECIFICATION' UNITS** — The coordinate values refer to user-specified increments of the display surface address space. These units are device-dependent and function similarly to the 'PERCENTUNITS' described above. 'SPECIFICATION UNITS' are defined as described in the next paragraph. Setting the 'SPECIFICATIONUNITS' to 100. in each direction results in 'SPECIFICATIONUNITS' which are identical to 'PERCENTUNITS'. The default units are 1000., along each display surface address space axis.

#### **SPECIFICATION UNIT SIZE:**

The number of specification units which occupy the display address space may be set by the following UPSET options. The default for each is 1000 units.

UPSET('SPECIFICATIONUNITS',value) — This sets the number of specification units along each axis of the display surface to the value specified.

UPSET('XSPECIFICATIONUNITS',value) — This sets the number of specification units along the X-axis of the display surface to the value specified. The number of units along the Y- and Z- axis remain unaffected.

UPSET('YSPECIFICATIONUNITS',value) — This sets the number of specification units along the Y-axis of the display surface to the value specified. The number of units along the X- and Z-axis remain unaffected.

UPSET('ZSPECIFICATIONUNITS',value) — This sets the number of specification units along the Z-axis of the display surface to the value specified. The number of units along the X- and Y-axis remain unaffected.

#### **COORDINATE SYSTEM:**

There are two types of coordinate systems available in GCS. One of these, the 'SYSTEM' or 'WORLD' coordinate system, is defined by GCS. The other, the 'USER' or 'MODELLING' coordinate system, is defined by the user. The 'USER' coordinate system is defined based on the 'WORLD' coordinate system or on other 'USER' coordinate systems. The 'USER' coordinate system can be cumulative or non-cumulative depending on whether the 'USER' coordinate system being created is to be a new 'REFERENCE' system or is to be a 'WORKING' system respectively. See U3CSYS and UCOSYS for a complete description of the 'USER' coordinate system facility. While 'USER' coordinate systems may be applied in 'DEVICE' space, the user is warned that a 'USER' coordinate system constructed in one coordinate space may not be correct if used in the other space.

**'SYSTEM' 'WORLD'** — If in 'DEVICE' space, GCS defines this coordinate system to refer to the unrotated, unscaled system inherent in the display surface addressing scheme. This normally places the origin at the lower left front corner of the display surface. If in 'VIRTUAL' space, this coordinate system is



unrotated with the origin located in virtual space at a position represented the computer system real numbers (0,0,0). The 'SYSTEM' or 'WORLD' coordinate system is the default setting. The 'WORLD' coordinate system may be USET to be either 'LEFTHANDED' or 'RIGHTHANDED'. Default is 'RIGHTHANDED'.

**'USER 'MODELLING'**

- The user may define his own coordinate system based on the currently specified coordinate system whether it be 'SYSTEM' or 'USER'. The default 'USER' coordinate systems are identical to the 'SYSTEM' coordinate system. GCS maintains two 'USER' coordinate system: a 'REFERENCE' system and a 'WORKING' system. These are defined below.

**USER COORDINATE SYSTEM TYPE:**

This specifies which of the two GCS-maintained 'USER' coordinate systems are to be used.

**'REFERENCE'**

- This selects the 'REFERENCE' user coordinate system. The effect of creating a new 'REFERENCE' system is cumulative as it is based on the previous 'REFERENCE' system. See UCOSYS and U3CSYS for details. This setting has no effect when using the 'SYSTEM' coordinate system.

**'WORKING'**

- This selects the 'WORKING' user coordinate system. The effect of creating a new 'WORKING' system is non-cumulative as it is based on the current 'REFERENCE' system. See UCOSYS and U3CSYS for details.

**LOGARITHMIC SCALING:**

This option allows the application of a logarithmic scale factor along any of the coordinate components. Any logarithmic base is supported and may be individually selected for each component (see below). Note that if the coordinate type is other than 'RECTANGULAR', the logarithmic scale factor may be applied to the angular unit components if desired (e.g., 'YLOGARITHMIC' would apply this factor to the THETA component of 'CYLINDRICAL' or 'POLAR' coordinates). See the Logarithmic Application Time options described below.

Application of a logarithmic scale factor will modify the size of the plot in relation to the window since the window is now looking at exponents rather than actual values. This may be adjusted for by some of the higher-level graphing routines (see U3AXIS).

**'LOGARITHMIC'  
'XYZLOGARITHMIC'**

- A logarithmic scale factor is applied along each component

**'NOLOGARITHMS'**

- Logarithmic scaling is disabled. This is the default.

**'XLOGARITHMIC'  
'YLOGARITHMIC'  
'ZLOGARITHMIC'  
'XYLOGARITHMIC'  
'XZLOGARITHMIC'  
'YZLOGARITHMIC'**

- The appropriate logarithmic scale factor is applied to the component indicated. Other components remain linear. 2-D applications should use 'XYLOGARITHMIC' to avoid conflicts with the default Z-value of 0.

### LOGARITHM APPLICATION TIME:

The effect of logarithmic scaling is dependent upon the state of the coordinates at the time the scaling is applied. Three options allow the user to direct the time at which the scaling is applied as follows:

- 'LOGOBJECT' — Logarithmic scaling is applied to the coordinates provided by the user after conversion to 'ABSOLUTE' but prior to any other transformations. In this mode, log scaling may be applied to the angle and/or radius components of 'CYLINDRICAL', 'POLAR', or 'SPHERICAL' coordinates.
- 'LOGUSER'  
'LOGMODELLING' — Logarithmic scaling is applied after conversion to 'ABSOLUTE', 'RECTANGULAR', 'USER' coordinates but before conversion to 'SYSTEM' coordinates. In this mode, logarithmic scaling will be applied along the current 'USER' coordinate system axes if 'USER' is specified. If 'SYSTEM' is specified, 'LOGUSER' is equivalent to 'LOGSYSTEM'.
- 'LOGSYSTEM'  
'LOGWORLD' — Logarithmic scaling is applied after conversion to 'SYSTEM' coordinates. In this mode, the logarithmic scaling will be applied along the 'SYSTEM' coordinate system axes. This is the default mode.

### LOGARITHM BASE:

The base to be used for logarithmic scaling can be selected for all coordinate components or individually for each coordinate component by using the following UPSET options. The value can be any positive real number or the character strings "E" or "e".

- UPSET('BASE',value) — The base for logarithmic scaling along each component is set to the value specified.
- UPSET('XBASE',value) — The base for logarithmic scaling along the X-component is set to the value specified. The other bases remain unaffected.
- UPSET('YBASE',value) — The base for logarithmic scaling along the Y-component is set to the value specified. The other bases remain unaffected.
- UPSET('ZBASE',value) — The base for logarithmic scaling along the Z-component is set to the value specified. The other bases remain unaffected.

### INTENSITY:

The brightness of the output on the display surface may be specified if the device has variable intensity settings. Intensity settings are specified by UPSET('BRIGHTNESS',value) where the value is a percentage along the range of allowable intensity settings from minimum to maximum. This 0. is the lowest intensity setting and 100. is the highest intensity setting. Intensities may also be specified by the following USET options. Default intensity level is 60%.

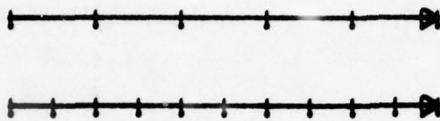
- 'DIM' — The intensity is set to the 10% level.
- 'NORMAL' — The intensity is set to the 60% level.
- 'BRIGHT' — The intensity is set to the 100% level.

#### **LINE WIDTH:**

The line width (sometimes referred to as spot size) may be specified if the device has a variable line width. Line widths are specified by UPSET ('WIDTH',value) where the value represents the width of the line in current units. The line width may also be set using the following USE 1 options. The default width is 'THIN'.

- 'THIN'           — The line width is set to the thinnest line. If a simulator is being used, a single line is generated.
- 'WIDE'           — The line width is set to the widest line. If a simulator is being used, a line approximately .1 inches in width is generated.

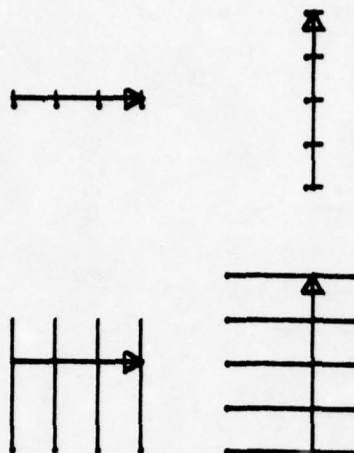




Tic Interval Specification  
Figure 2 (USPEN)

```
CALL USTART
CALL UPSET ('TERMINATOR','')
CALL USET ('TARROW')
CALL UMOVE (0.,50.)
CALL UPSET ('TICINTERVAL',10.)
CALL UPEN (50.,50.)
CALL UMOVE (0.,40.)
CALL UPSET ('TICINTERVAL',5.)
CALL UPEN (50.,40.)
CALL USET ('ACENTER')
CALL USET ('LARGE')
CALL UPRINT (25.,30., 'T>IC <I>NTERVAL <S>PECIFICATION<,'')
CALL UPRINT (25.,25., 'F>IGURE 2 <(USPEN),')
CALL UEND
STOP
END
```





Tic Mark Sizes  
Figure 3 (USPEN)

```

CALL USTART
CALL UPSET ('TERMINATOR',',','')
CALL USET ('TARROW')
CALL UPSET ('TICINTERVAL',5.)
CALL UMOVE (5.,78.)
CALL UPEN (28.,78.)
CALL UMOVE (48.,88.)
CALL UPEN (48.,88.)
CALL UPSET ('TICPLUS',10.)
CALL UPSET ('TICMINUS',5.)
CALL UMOVE (5.,48.)
CALL UPEN (28.,48.)
CALL UMOVE (48.,38.)
CALL UPEN (48.,58.)
CALL USET ('ACENTER')
CALL USET ('LARGE')
CALL UPRINT (25.,15., 'T>IC <N>ARK <S>IZES,')
CALL UPRINT (25.,18., '<F>IGURE 3 <USPEN>,'')
CALL UEND
STOP
END

```

```

58_ _ _ _ _
18. . . . .
74_ _ _ _ _
744_ _ _ _ _
3454_ _ _ _ _
7282_ _ _ _ _
12345678_ _ _ _ _
1_ _ _ _ _ (NON-HARDWARE DASH)
8 . . . . . (NON-HARDWARE DASH)

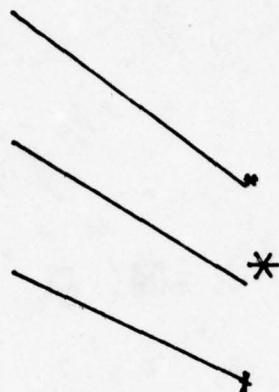
```

Examples of Dashed Lines  
Figure 4 (USPEN)

```

DIMENSION DTABLE(8)
DATA DTABLE/8.,1.,12345678.,7282.,3454.,744.,74.,18.,58./
CALL USTART
CALL UPSET ('TERMINATOR','(',')')
CALL USET ('LARGE')
CALL USET ('INTEGER')
CALL USET ('DNULL')
DO 18 I = 38,78,5
Y = I
J = (I-38) / 5 + 1
CALL UPSET ('SETDASH',DTABLE(J))
CALL UPRINT (8.,Y,DTABLE(J))
18 CALL UPEN (58.,Y)
CALL USET ('TEXT')
CALL UPRINT (58.,35.,'(NON-HARDWARE DASH),')
CALL UPRINT (58.,38.,'(NON-HARDWARE DASH),')
CALL USET ('ACENTER')
CALL UPRINT (58.,28.,'(E)XAMPLES OF (D)ASHED (L)INES,')
CALL UPRINT (58.,15.,'(F)IGURE 4 (USPEN),')
CALL UEND
STOP
END

```



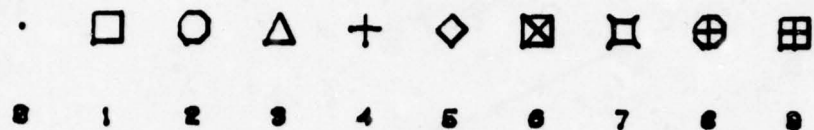
Character Terminator Output

Figure 5 (USPEN)

```

CALL USTART
CALL UPSET ('TERMINATOR','')
CALL USET ('PERCENT UNITS')
CALL USET ('LCHARACTERS')
CALL USMOVE (10.,60.,0.)
CALL USPEN (50.,60.,-50.)
CALL USET ('SIMULATED HARDWARE CHARACTERS')
CALL UPSET ('XSIZE',5.)
CALL UPSET ('YSIZE',7.)
CALL USMOVE (10.,65.,0.)
CALL USPEN (50.,65.,-50.)
CALL USET ('SOFTWARE CHARACTERS')
CALL USMOVE (10.,50.,0.)
CALL USPEN (50.,50.,-50.)
CALL USET ('ACENTER')
CALL USET ('HARDWARE CHARACTERS')
CALL USET ('LARGE')
CALL USPRINT (25.,30.,0., 'C>HARACTER <T>ERMINATOR <O>UTPUT,')
CALL USPRINT (25.,25.,0., 'F>IGURE 5 <U>SPEN,')
CALL UEND
STOP
END

```



GCS-Defined Symbols

Figure 6 (USPEN)

```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('NSYMBOL')
CALL USET ('ACENTER')
CALL USET ('INTEGER')
CALL USET ('LARGE')
CALL UPSET ('SZMARKER',.3)
DO 18 I = 5,95,18
X = I
SYM = (I-5) / 18
CALL UPSET ('SYMBOL',SYM)
CALL UPEN (X,99.)
18 CALL UPRINT (X,59.,SYM)
CALL USET ('TEXT')
CALL UPRINT (59.,49., 'GCS-DEFINED <S>YMBOLS,')
CALL UPRINT (59.,35., '<F>IGURE 6 <(USPEN),')
CALL UEND
STOP
END

```



\_\_\_\_\_ (30.,60.)

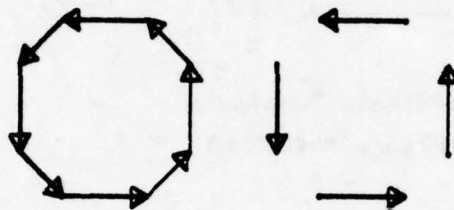
\_\_\_\_\_ (50..50..0.)

\_\_\_\_\_ (30..40.)

### Coordinate Terminators

Figure 7 (USPEN)

```
CALL USTART
CALL UPSET ('TERMINATOR',',')
CALL USET ('LCOORDINATES')
CALL USET ('LARGE')
CALL UMOVE (0.,60.)
CALL UPEN (30.,60.)
CALL USET ('SIMULATED HARDWARE')
CALL USET ('3DCOORDINATES')
CALL UMOVE (0.,50.)
CALL UPEN (30.,50.)
CALL USET ('SOFTWARE')
CALL USET ('2DCOORDINATES')
CALL UMOVE (0.,40.)
CALL UPEN (30.,40.)
CALL USET ('ACENTER')
CALL USET ('HARDWARE')
CALL UPRINT (30.,30., 'C>COORDINATE <T>ERMINATORS,')
CALL UPRINT (30.,25., '<F>IGURE 7 <USPEN,')
CALL UEND
STOP
END
```

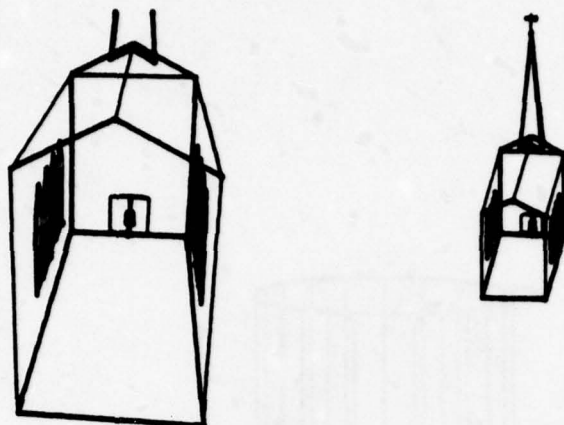


Gapped Lines  
Figure 8 (USPEN)

```

CALL USTART
CALL UPSET ('TERMINATOR',',','')
CALL USET ('LARRROW')
CALL USET ('UNINTERRUPTED')
ILOOP = 0
XLOC = 25.
18 IF (ILOOP .EQ. 2) GO TO 28
CALL UMOVE (XLOC+5.,38.)
CALL UPEN (XLOC+15.,38.)
CALL UPEN (XLOC+28.,35.)
CALL UPEN (XLOC+28.,45.)
CALL UPEN (XLOC+15.,58.)
CALL UPEN (XLOC+5.,58.)
CALL UPEN (XLOC,45.)
CALL UPEN (XLOC,35.)
CALL UPEN (XLOC+5.,38.)
ILOOP = ILOOP + 1
XLOC = 55.
CALL USET ('GAPPED')
GO TO 18
28 CALL USET ('ACENTER')
CALL USET ('LARGE')
CALL UPRINT (58.,28., 'GAPPED <L>INES,')
CALL UPRINT (58.,15., '<F>IGURE 8 <USPEN>,'')
CALL UEND
STOP
END

```

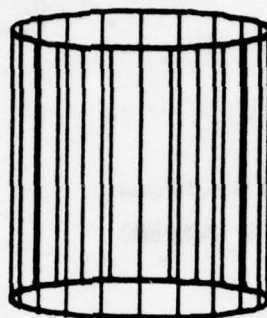


Projection Modes  
Figure 9 (USPEN)

```

CALL ATTACH (2, '/TEKSDENO/SAVE', '3,8,1ST,')
CALL USTART
CALL UPSET ('TERMINATOR', ',,')
CALL UASPCT (1.)
CALL UPSET ('LIBRARYFILE', 1.)
CALL UTILITY ('LOAD', 2.)
CALL USET ('ACENTER')
CALL USET ('LARGE')
CALL UPRINT (50., 25., 'P>ROJECTION <MODES,')
CALL UPRINT (50., 20., '<F>IGURE 9 <(USPEN),')
CALL USET ('NOCENTER')
CALL USET ('REFERENCE SYSTEM')
CALL UVIEW (-40., 200., 70., -20., 20., 0.)
CALL UWINDO (-100., 100., -100., 100.)
CALL UVMPRT (150.)
CALL USET ('PERSPECTIVE')
CALL USET ('PERCENT UNITS')
CALL UDAREA (0., 50., 25., 75.)
CALL U3CALL (-50., 20., 0., 1., 1., 1., 00., 0., 0., 'CHURCH')
CALL USET ('ORTHOGRAPHIC')
CALL UDAREA (50., 100., 25., 75.)
CALL U3CALL (-50., 20., 0., 1., 1., 1., 00., 0., 0., 'CHURCH')
CALL UEND
STOP
END

```



Cylindrical Coordinates

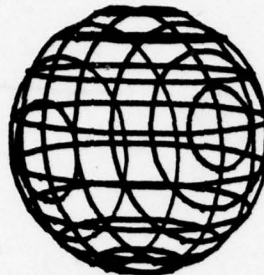
Figure 18 (USPEN)

```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('LARGE')
CALL USET ('ACENTER')
CALL UPRINT (50.,20., 'CYLINDRICAL COORDINATES,')
CALL UPRINT (50.,15., 'FIGURE 18 (USPEN),')
CALL UVIEW (150.,150.,30.,0.,0.,30.)
CALL UWINDO (-100.,100.,-100.,100.)
CALL USET ('CYLINDRICAL')
CALL UCRLE (0.,0.,30.)
CALL UPSET ('ZVALUE',00.)
CALL UCRLE (0.,0.,30.)
DO 10 I=1,340,15
  THETA = I - 1
  CALL USMOVE (30.,THETA,0.)
10 CALL USPEN (30.,THETA,00.)
CALL UEND
STOP
END

```



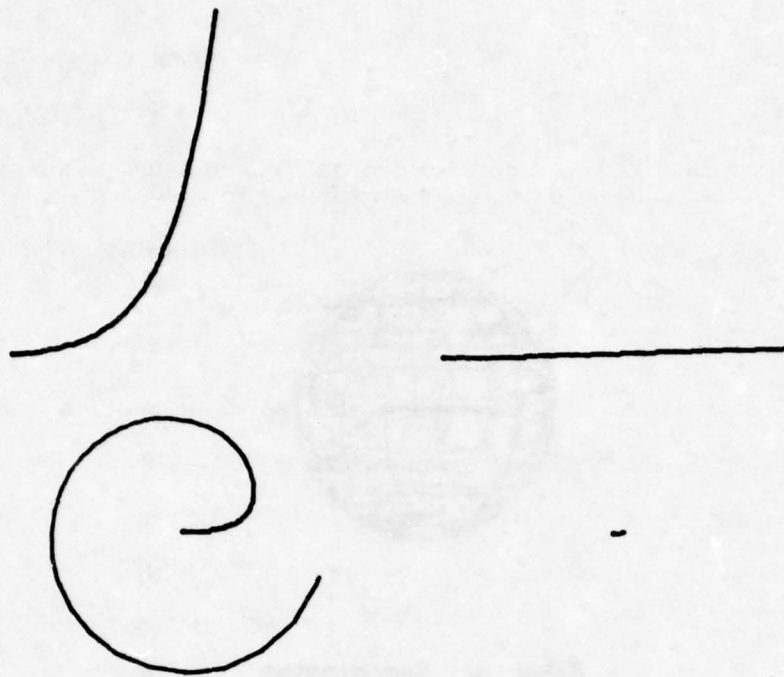


Spherical Coordinates  
Figure 11 (USPEN)

```

CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('ACENTER')
CALL USET ('LARGE')
CALL UPRINT (50.,20., 'SPHERICAL <C>COORDINATES,')
CALL UPRINT (50.,15., '<F>IGURE 11 <C>USPEN,')
CALL UVIEW (150.,150.,30.,0.,0.,0.)
CALL UWINDO (-100.,100.,-100.,100.)
CALL USET ('SPHERICAL COORDINATES')
RADIUS = 30.
ILOOP = 0
10 IF (ILOOP .EQ. 2) GO TO 20
DO 15 I = 1, 181, 20
  RHO = I - 1
  CALL USMOVE (RADIUS,0.,RHO)
  DO 15 J = 1, 361, 10
    THETA = J - 1
15 CALL USPEN (RADIUS,THETA,RHO)
  CALL USMOVE (0.,0.,0.)
  CALL USWOTA (0.,90.,0.)
  ILOOP = ILOOP + 1
  GO TO 10
20 CONTINUE
CALL UEND
STOP
END

```



# Logarithmic Scaling

Figure 12 (USPEN)

```

DIMENSION X(101),Y(101)
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UASPCT (1.)
DO 10 I = 1, 101
  X(I) = I
10 Y(I) = 10.00(X(I)/90.)
  CALL USET ('PERCENT UNITS')
  CALL UDAREA (5.,45.,60.,100.)
  CALL UMOVE (X(I),Y(I))
DO 20 I = 2, 101
20 CALL UPEN (X(I),Y(I))
  CALL USET ('YLOGARITHMIC')
  CALL UDAREA (55.,95.,60.,100.)
  CALL UMOVE (X(I),Y(I))
DO 30 I = 2, 101
30 CALL UPEN (X(I),Y(I))
  DO 40 I = 1, 101
    X(I) = FLOAT(I) / 1.2
40 Y(I) = 2.00(X(I) / 10.)
  CALL USET ('POLAR')
  CALL UWINDO (-100.,100.,-100.,100.)
  CALL USET ('NOLOGARITHMS')
  CALL UDAREA (5.,45.,20.,60.)
  CALL UMOVE (X(I),Y(I))
DO 50 I = 2, 101
50 CALL UPEN (X(I),Y(I))
  CALL UDAREA (55.,95.,20.,60.)
  CALL USET ('XYLOGARITHMIC')
  CALL UPSET ('BASE',2.)

```

```

CALL USET ('LOGOJECT')
CALL UMOVE (X(1),Y(1))
DO 99 I = 2, 101
99 CALL UPEN (X(I),Y(I))
CALL USET ('NOLOGARITHMS')
CALL USET ('RECTANGULAR')
CALL UHINDO (0.,100.,0.,100.)
CALL UDAREA (0.,100.,0.,100.)
CALL USET ('LARGE')
CALL USET ('ACENTER')
CALL UPRINT (50.,20., 'LOGARITHMIC <S>CALING,')
CALL UPRINT (50.,15., '<F>IGURE 12 <USPEN>')
CALL UEND
STOP
END

```

**FUNCTION:**

This routine sets the USET option provided, draws the indicated 3D line using the current line option, and then restores the value changed by the USET option.

**CALLING SEQUENCE:**

**CALL U3PEN1(X,Y,Z,OPTION)**

**Where**

**X** is the X (RADIUS) coordinate of the head of the beam/pen movement  
**Y** is the Y (THETA) coordinate of the head of the beam/pen movement  
**Z** is the Z (PHI) coordinate of the head of the beam/pen movement  
**OPTION** is the USET option to be set only for the execution of the subroutine

**OPTIONS which may apply:**

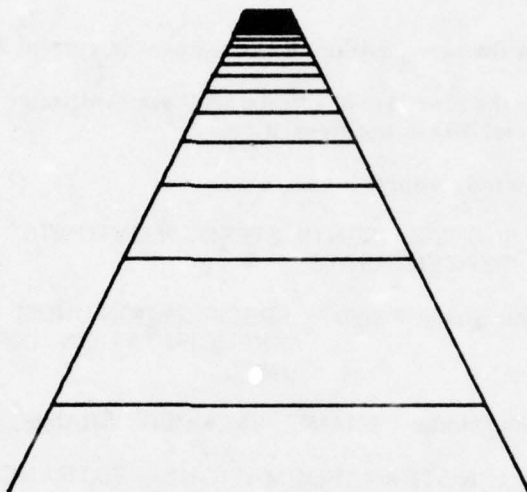
Any USET option.  
See UPEN/U3PEN for a description of options effective pen/beam movements.

**COMMENTS:**

U3PEN1 is convenient for setting options which will apply only to one pen/beam movement. The GSA setting modified by OPTION will be restored prior to returning to the calling program.

**Programming Notes:**





```

CALL USTART
CALL UVIEW (0.,0.,10.,0.,10.,10.)
CALL UWINDO (-10.,10.,-12.,6.)
CALL USMOVE (5.,5.,0.)
CALL USPEN (5.,100.,0.)
DO 10 I = 10, 100, 5
CALL USMOVE (-5.,FLOAT(I-5),0.)
CALL USPEN (-5.,FLOAT(I),0.)
10 CALL USPEN1 (10.,0.,0., 'RELATIVE')
CALL UEND
STOP
END

```

**FUNCTION:**

This routine applies a 3-D image translation transformation to the indicated segment/frame.

**CALLING SEQUENCE:**

**CALL U3PLAC (X,Y,Z,SEGID)**

Where

**X,Y,Z** is the new position of the segment in current 3-D device units.

**SEGID** is the identifier of a "retained" segment/frame which was UOPENed for at least 3-D image translations.

**OPTIONS which may apply:**

Device Units: 'INCHES', 'CENTIMETERS', 'FONTUNITS', 'SPECIFICATION UNITS', 'PERCENT UNITS'

Specification Unit Size (UPSET): 'SPECIFICATION UNITS', 'XSPECIFICATION UNITS', 'YSPECIFICATION UNITS', 'ZSPECIFICATION UNITS'

Segment Identifier Mode: 'FNAME', 'FNUMBER', 'SNAME', 'SNUMBER'

Segment Type: 'NOTTRANSFORMATIONS', '2DTRANSLATION', '2DGENERAL', '3DTRANSLATION', '3DGENERAL'

**COMMENTS:**

The image transformation is applied to the specified segment. If the resulting image exceeds the display dimensions, the result is undetermined.

Image transformations are only applied if supported on the current display surface. Requests for image transformations will be ignored if the display device does not support this facility.

**Programming Notes:**

**FUNCTION:**

This routine provides a general-purpose numeric 3-D plotting capability. Given three arrays of corresponding coordinates of one or more curves, it will scale and plot these points along with suitable axes and labels as specified by the user.

**CALLING SEQUENCE:**

**CALL U3PLOT(X,Y,Z,CURVES,PTS,OPTS)**

Where

- X** is a set of X or RADIUS COMPONENTS for the points for all the curves in current user units.
- Y** is a set of Y or THETA components for the points for all the curves in current user units.
- Z** is a set of Z or PHI components for the points for all the curves in current user units.
- CURVES** is a single variable which indicates the number of curves to be plotted.
- PTS** is an array which indicates how many points are in each curve.
- OPTS** is an array which specifies which USET option will apply to each curve as it is being plotted. One option must be specified for each curve and only the first four characters of the option name should be specified.

**OPTIONS which may apply:**

All which apply to U3PLOT  
All which apply to U3AXIS

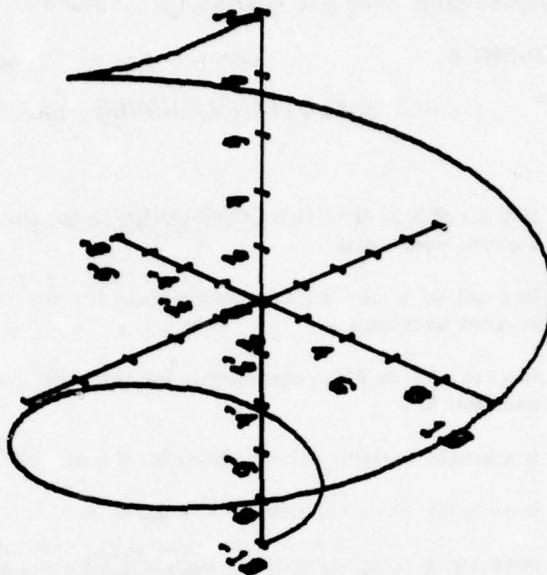
X-Component Repetition Option: 'XREPEAT', 'NOXREPEAT', 'XCONSTANT'  
Y-Component Repetition Options: 'YREPEAT', 'NOYREPEAT', 'YCONSTANT'  
Z-Component Repetition Options: 'ZREPEAT', 'NOZREPEAT', 'ZCONSTANT'

**COMMENTS:**

Since the U3AXIS options apply, the user should become familiar with their effects.

The Component Repetition Options allow the user to avoid repetitious storing of components which retain the same value throughout a particular curve or over a group of curves. Selecting a 'CONSTANT' option indicates that that component does not vary during any of the drawing. Selecting a 'NOREPEAT' option indicates that a component is provided for every point in every curve. This is the default. Selecting a 'REPEAT' option indicates that one set of component values is provided which will be reused for each curve. The size of this set is always the number of points in the first curve. If subsequent curves have more points than the first curve, U3PLOT will start to resequence through the 'repeated' components.

**Programming Notes:**



SPIRAL ON A SPHERE

```

REAL X(401), Y(401), Z(401), R
R = 10.
DO 10 I = 1, 401
  Z(I) = FLOAT(I-1) * R/200.0 - R
  T = SORT(R**2 - Z(I)**2)
  X(I) = COS(Z(I) * 0.5) * T
  Y(I) = SIN(Z(I) * 0.5) * T
10 CONTINUE
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL UPSET ('HORIZONTAL',.6)
CALL UPSET ('VERTICAL',1.2)
CALL USET ('SOFT')
CALL USET ('NOYLABELS')
CALL UWINDO (-15.,15.,-15.,15.)
CALL UVIEW (30.,-30.,20.,0.,0.,0.)
CALL USPLOT (X,Y,Z,1.,401.,'LNULL')
CALL UVIEW (0.,0.,150.,0.,0.)
CALL USET ('ACENTER')
CALL USMOVE (0.,-11.,0.)
CALL UPRNT1 ('SPIRAL ON A SPHERE',',','TEXT')
CALL UEND
STOP
END

```



**FUNCTION:**

This routine displays textual data justified at the 3D pen position specified. At termination, the pen will be positioned at the end of the text string ready for additional characters.

**CALLING SEQUENCE:**

**CALL U3PRNT (X,Y,Z,DATA)**

**Where**

**X,Y,Z** are the coordinates of the text string justification positioned in current units.

**DATA** is a GCS Hollerith text string if 'TEXT' is specified.  
is a REAL variable if 'INTEGER' or 'REAL' is specified.  
is a REAL array of two elements if 'XYCOORDINATES' are specified.  
is a REAL array of three elements if 'XYZCOORDINATES' are specified.

**OPTIONS which may apply:**

Alphanumeric Output Type: 'TEXT', 'INTEGER', 'REAL', 'XYCOORDINATES', 'XYZCOORDINATES'.

GCS Terminator Character: UPSET('TERMINATOR',value)

Numeric Precision Option: UPSET('PRECISION',value)

Significant Zero Option: 'SIGNIFICANT ZEROES', 'NOSIGNIFICANTZEROES'

Character Type Options: 'HARDWARE', 'SIMULATED HARDWARE', 'SOFTWARE'

Hardware Character Position Size: 'SMALL', 'MEDIUM', 'LARGE', 'EXTRALARGE',  
UPSET('XSIZE',value), UPSET('YSIZE',value)

Software Character Position Size: UPSET('HORIZONTALSIZE',value), UPSET('VERTICALSIZE',value)

Character Position Occupancy: UPSET('XPERCENT',value), UPSET('YPERCENT',value), UPSET('ZPERCENT',value)

Alphabetic Case Shifting: 'UPPERCASE', 'LOWERCASE'

Case Shift Character Specification: UPSET('UPPERCASE',value),  
UPSET('LOWERCASE',value)

Orientation: UPSET('ORIENTATION',value)

Italicization Options: 'ITALICS', 'NOITALICS'

Italic Slant Angle: UPSET('SLANT',value)

Coordinate Space Options: 'VIRTUAL', 'DEVICE'

Margin Boundaries: see UMARGN

Window Boundaries: see UWINDO

Alphanumeric Spacing Options: 'VERTICAL', 'HORIZONTAL'

Alphanumeric Justification Options: 'NOCENTERING', 'ACENTERING',  
'NOJUSTIFICATION', 'LJUSTIFICATION',  
'CJUSTIFICATION', 'RJUSTIFICATION'

Subscript Character: UPSET('SUBSCRIPT',value)

Superscript Character: UPSET('SUPERScript',value)

Subscript/Superscript Specification: 'SUBSCRIPT', 'SUPERScript', 'NOScript'

Scripting Level: UPSET('SCRIPTLEVEL',value)

Pen Coordinate Option: see U3PEN

Coordinate System: 'SYSTEM', 'WORLD', 'USER', 'MODELLING'

User Coordinate System Type: 'REFERENCE', 'WORKING'

Attributes: see U3PEN

#### **COMMENTS:**

The description of the alphanumeric output facilities is divided into three areas of format character descriptions and alphanumeric output, positioning each of these will be discussed individually.

#### **Programming Notes:**

## **FORMAT:**

U3PRNT handles any of five types of alphanumeric operations. The format of the input parameter DATA depends upon which type of operation is specified.

- 'INTEGER'**
- The input is a single real variable whose integer value will be displayed at (X,Y,Z). Truncation will occur if a number which is not an integer is provided.
- 'REAL'**
- The input is a single real number whose value will be displayed at (X,Y,Z). The number of digits of precision can be adjusted by using UPSET ('PRECISION',value) where the value is the number of digits of precision desired. Numbers will be rounded to the least significant digit. The default precision is four. Display of significant trailing zeroes is controlled by the significant zero USET option. 'SIGNIFICANT ZEROES' specifies that significant zeroes are to be displayed; 'NOSIGNIFICANT ZEROES' suppresses their display. The default suppresses significant zeroes.
- 'XYCOORDINATES'**
- The input is a real array of two elements. The first element reflects an X- or RADIUS coordinate component at the second element. The second reflects a Y- or THETA coordinate component. The coordinates will be displayed at (X,Y,Z) enclosed in parentheses and separated by a comma.
- 'XYZCOORDINATES'**
- The input is a real array of three elements. The first element reflects an X- or RADIUS coordinate component; the second element, a Y- or THETA coordinate component; and the third element, a Z- or PHI coordinate component. The coordinates will be displayed at (X,Y,Z) enclosed in parentheses and separated by commas; e.g., (3.8765,1.5,0.). The precision and significant zeroes options, described for 'REAL' numbers above, both apply to each component. Note that the coordinates displayed need not be the same values as the alphanumeric output justification position parameters (X,Y,Z).
- 'TEXT'**
- The input is a single variable or array containing a Hollerith Character string. The string may be as long as desired but must be terminated by the GCS terminator character. This character, which cannot be displayed while functioning as the terminator character, may be specified by UPSET ('TERMINATOR', value) where value is a one-character Hollerith string. The default GCS terminator character is the ASCII backslash (12-8-5) character. The input string may also contain upper and lower case shifting characters. The case shifting and scripting characters are described below.

## **CHARACTER DESCRIPTION OPTIONS:**

These options specify the attributes of individual characters.



## CHARACTER TYPE:

There are three basic types of characters in GCS. Each type will be described separately below. Figure 2 illustrates each type. The user can switch between any of the three types simply by invoking the appropriate USET options.

### 'HARDWARE'

- These characters are generated when possible by a hardware character generator in the display device. The size of these character positions is frequently limited to several discrete sizes (see Hardware Character Position Size below). The characters are produced on the display surface plane. This is the default character type. Character spacing is exact. Hardware character output may be directed to either the 'MESSAGE DEVICE' or the 'PLOTDEVICE' if they are separable.

### 'SIMULATED HARDWARE'

- These characters are produced via software. Although they act as hardware characters, they are generated to the exact hardware character position size (q.v.) specified. 'SIMULATED HARDWARE' characters use the 'SOFTWARE' characters descriptions. The characters are produced on the display surface plane. Character spacing is exact.

### 'SOFTWARE'

- These characters are produced by software vectors in the current XY-plane. The line which connects the lower left corner of each character position in the string is parallel to the current X-axis. The string is drawn in the positive X-direction for horizontal spacing and the negative Y-direction for vertical spacing (see Character spacing). Both software character dimensions may be specified (see below).

### HARDWARE CHARACTER POSITION SIZE

- The size of 'HARDWARE' and 'SIMULATED HARDWARE' character positions may be set by either USET or UPSET options. An exact size may be specified by the following UPSET options. For hardware characters, the largest discrete size will be used which does not exceed the requested size. If all sizes exceed the requested size, then the smallest size is used.

UPSET('XSIZE',value) — The horizontal character position size is set to the value specified in current 'DEVICE' space units.

UPSET('YSIZE',value) — The vertical character position size is set to the value specified in current 'DEVICE' space units.

Four (4) discrete sizes are provided by USET options:

- 'SMALL' — The hardware character position size closest to typewriter size character positions. This is the default.
- 'MEDIUM' — This is the next larger size from 'SMALL'. Simulated 'MEDIUM' size is approximately twice the 'SMALL' size.
- 'LARGE' — This is the next larger size from 'MEDIUM'. Simulated 'LARGE' size is approximately three times the 'SMALL' size.



**'EXTRALARGE'** — This is the next larger size from 'LARGE'. Simulated 'EXTRALARGE' size is approximately four times the 'SMALL' size.

Additional discrete sizes may be specified by:

**UPSET('SIZE',value)** — The discrete size whose index is specified in the value is selected. Indexes must be positive values. 'SMALL' through 'LARGE' are represented by index values 1. through 4. Assignment of sizes to remaining index values (i.e.,LT.4.) is installation dependent.

#### **SOFTWARE CHARACTER POSITION SIZE:**

The size of 'SOFTWARE' character positions may be set by the following two UPSET options. The visible size of software characters is affected by scaling factors created by U3CSYS calls and by the currently set viewing environment. Software character positions in device space are visibly the same size as software character positions in virtual space at the view plane.

**UPSET('HORIZONTAL SIZE',value)** — The horizontal size of software character positions is set to the value provided in current user units. Default value is 5.

**UPSET('VERTICALSIZE',value)** — The vertical size of software character positions is set to the value provided in current user units. Default value is 7.

#### **CHARACTER POSITION OCCUPANCY:**

The proportion of the software character positions to actually be occupied by the character may be specified as follows:

**UPSET('XPERCENT',value)** — The proportion of the width of the character position to be occupied by the character is set to the value provided. Valid values are greater than zero. A value of 1. specifies the entire width of the character position. Default value is .65.

**UPSET('YPERCENT',value)** — The proportion of the height of the character position to be occupied by the character is set to the value provided. Valid values are greater than zero. A value of 1. specifies the entire height of the character position. Default value is .65.

#### **ALPHABETIC CASE SHIFTING:**

In machines which have only six bits per character, it is still desirable in GCS to be able to specify both upper and lower case characters since the software character set can produce both. Therefore, a case shifting facility has been implemented which not only provides this service for the software characters but will also work for hardware characters on those devices which have hardware character generators which can produce both cases. Additionally, since case shifting applies only to upper case characters, when a program which does case shifting is executed on a computer which does not need case shifting, the desired upper and lower case character will still be produced.

To shift cases, it is necessary to insert special case shift characters in the GCS text string. GCS defaults to upper case. When the lower case shift character inserted in the

string, a shift to lower case will occur if GCS is currently in upper case. If GCS is already in lower case, the shift character will itself be generated. The opposite happens when the upper case shift character is inserted in the string. If GCS is in lower case mode, it will be placed in upper case. However, if GCS is already in upper case mode, the upper case shift character will be generated. See Figure 3 for an illustration of the use of case shifting.

When changing cases, all alphabetic characters will be generated in either UPPERCASE or LOWERCASE as appropriate. However, to provide the full complement of special characters available in the ASCII character set to persons with computers which handle only 6-bits per character internally, some special characters will be mapped into others when the case is changed.

The shift characters may be changed by the following two calls:

```
CALL UPSET('LOWERCASECHARACTER',CHAR)
or
CALL UPSET('UPPERCASECHARACTER',CHAR)
```

Where

CHAR is a single character in Hollerith (left-justified, blank-fill) format. Thus, it may be a literal (quoted character string) or a variable containing a Hollerith value.

The user is warned that the shift character should not be set to be the line terminator character. If this happens, no case shifting will occur. The case in which GCS is currently set may be forced to either upper or lower case by the following:

```
CALL USET('UPPERCASE')
CALL USET('LOWERCASE')
```

This is convenient for setting the system character for use as CHARACTER terminators in the line option or in building ALPHA lines.

#### ORIENTATION:

As software characters are produced, they may be 'ORIENTED' (tilted) off the X-axis by the UPSET option described below. For a string of characters emanating from U3PRNT, the lower left corner of each character will touch a line which is parallel to the current X-axis. The bottom of the character will be rotated off this line by the number of angular units specified. An illustration of this feature is shown in Figure 4.

UPSET('ORIENTATION',value) — The geometric figure orientation parameter is set to the number of angular units specified in the value. Default value is 0. Note that the orientation parameter also applies to UPLYGN and URECT.

#### ITALIZATION:

Software characters may be 'ITALICIZED' by applying transformation which leans or slants the character away from the vertical as illustrated in Figure 5. This mode is selected as follows:

'ITALICS' — Software characters are to be slanted.  
'NOITALICS' — Software characters are not to be slanted. This is the default.

### ITALIC SLANT ANGLE:

The amount of slant to be applied to the characters may be specified by:

UPSET('SLANT',value) — The amount of slant from the vertical is set to the value provided in current angular units. Default value is approximately 17.5 degrees. Values provided must not be odd multiples of 90 degrees.

### COORDINATE SPACE OPTIONS:

U3PRNT can display characters in either 'DEVICE' or 'VIRTUAL' space. If 'DEVICE' space is specified, all characters will be subject to margining; i.e., all characters will be produced within the specified margins (see UMARGN).

If 'VIRTUAL' space is specified, all characters will be clipped at the window boundaries (see UWINDO). For 'HARDWARE' and 'SIMULATED HARDWARE' characters, clipping will occur if any portion of the character position exceeds the window boundaries. Since 'SOFTWARE' characters are drawn with vectors, they will be clipped exactly at the window boundaries.

### ALPHANUMERIC SPACING OPTIONS:

Adjacent character positions may be defined to proceed either horizontally or vertically. This option applies to all character types and is selected as follows:

'HORIZONTAL' — Adjacent character positions occur horizontally; i.e., character strings are produced in the positive X direction.

'VERTICAL' — Adjacent character positions occur vertically; i.e., character strings are produced in the negative Y direction.

### ALPHANUMERIC JUSTIFICATION OPTIONS:

The arguments to U3PRNT contain a set of coordinates which indicate the justification position for the output string. Each justification option is illustrated in Figure 9. Justification options are as follows:

'LJUSTIFICATION'  
'NOCENTERING'  
'NOJUSTIFICATION' — The coordinates specify the position of the lower left corner of the first character position in the output string. This is the default.

'CJUSTIFICATION'  
'ACENTERING' — The coordinates specify the position of the center of the output string. The output will be centered both horizontally and vertically. The centering is upon the unscripted string (see Subscript/Superscript Specification below).

'RJUSTIFICATION' — The coordinates specify the position of the lower left corner of the character position which follows the last character of the string. Note that this is also the lower right corner of the last character position of the string if 'HORIZONTAL' spacing is specified.

### SUBSCRIPT/SUPERSCRIPIT SPECIFICATION:

The primary means of doing scripting is by the insertion in GCS text strings of subscript and superscript escape characters. Each occurrence of the subscript character will lower the position of succeeding characters 1/3 character position. Each occurrence of the superscript character will raise the position of succeeding characters 1/3 character



position. When the GCS terminator character is encountered, the scripting level is returned to the zero position and the pen/beam position is set for additional unscripted characters. (See Figure 10.) Note that scripting positions will not be computed into center justification adjustments. The subscript and superscript escape characters may be specified by the following UPSET options:

UPSET('SUBSCRIPT',value) — The subscript escape character is set to the first character of the Hollerith string provided as a value.  
The default subscript character is an ASCII " " (0/8/5).

UPSET('SUPERScript',value) — The superscript escape character is set to the first character of the Hollerith string provided as a value.  
The default superscript character is an ASCII " " (0/8/5).

An alternate method of performing scripting is provided if several successive output strings are to be scripted. The user may specify the type of scripting and the scripting level. The scripting level is defined to be the number of 1/3 character positions below the justification position for 'SUBSCRIPTING' and the number of 1/3 character positions above the justification position for 'SUPERScriptING'.

'NOSCRIPTING' — Forced scripting is disabled. However, scripting by insertion of scripting escape characters can still occur.  
This is the default.

'SUBSCRIPTING' — The output string will be produced starting at the specified script level position below the justification position.

'SUPERScriptING' — The output string will be produced starting at the specified script level position above the justification position.

Note that specification of the type of scripting still allows further modifications in the scripting by insertion of scripting escape characters. To set the scripting level, the following UPSET option should be used:

UPSET('SCRIPTLEVEL',value) — The scripting level is set to the integer value specified.  
Note that if a negative scripting level is provided, subscripting becomes superscripting and vice versa.

#### **PEN COORDINATE OPTIONS:**

All pen coordinate options apply to the justification position contained in the X,Y,Z parameters to U3PRNT. The actual starting location of the character string may be adjusted as required by the justification option described earlier. For a full description of the Pen Coordinate Options, see U3PEN.

#### **COORDINATE SYSTEM SELECTION AND TYPE:**

The full power of U3PRNT can only be realized through use of user coordinate systems. U3PRNT produces 'SOFTWARE CHARACTERS' on the XY-plane which passes through the justification point. Through use of user coordinate systems, the user can position and orient the XY-plane anywhere in 3-space. By suitable coordinate system definitions, 'SOFTWARE' character output can be made to recede into the distance or be viewed from behind. It should be noted that while 'HARDWARE' and 'SIMULATED HARDWARE' characters can be positioned anywhere in 3-space, they will still be produced in the display surface plane.



123456

3.142

(3.142,2.718)

(3.142,2.718,10.)

Example of 'TEXT' Output

#### Alphanumeric Output Formats

Figure 1 (USPRINT)

```
DIMENSION COORD(3)
DATA COORD/3.1415926,2.71828,10./
CALL USTART
CALL UPSET ('TERMINATOR',' ','')
CALL USET ('LARGE')
CALL USET ('INTEGER')
CALL UPRINT (28.,50.,123456.)
CALL USET ('REAL')
CALL UPRINT (28.,45.,COORD(1))
CALL USET ('XYCOORDINATES')
CALL UPRINT (28.,40.,COORD)
CALL USET ('XYZCOORDINATES')
CALL UPRINT (28.,35.,COORD)
CALL USET ('TEXT')
CALL UPRINT (28.,30., 'EXAMPLE OF <'TEXT'> OUTPUT,')
CALL USET ('ACENTER')
CALL UPRINT (40.,20., '<A>ALPHANUMERIC <O>UTPUT <F>ORMATS,')
CALL UPRINT (40.,15., '<F>IGURE 1 <USPRINT>,')
CALL UEND
STOP
END
```

## Hardware Characters

# Simulated Hardware Characters Software Characters

### GCS Character Types

Figure 2 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',',')
CALL USET ('ACENTER')
CALL UROTAT (18.)
CALL USET ('EXTRALARGE')
CALL UPRINT (50.,50., 'H>ARDWARE <C>HARACTERS,')
CALL USET ('SIMULATED HARDWARE CHARACTERS')
CALL UPSET ('XSIZE',.25)
CALL UPSET ('YSIZE',.4)
CALL UPRINT (50.,40., '<S>IMULATED <H>ARDWARE <C>HARACTERS,')
CALL USET ('SOFTWARE CHARACTERS')
CALL UPSET ('HORIZONTAL SIZE',4.5)
CALL UPSET ('VERTICAL SIZE',7.)
CALL UPRINT (50.,25., '<S>OFTWARE <C>HARACTERS,')
CALL USET ('HARDWARE CHARACTERS,')
CALL USET ('LARGE')
CALL UPRINT (45.,15., '<GCS C>HARACTER <T>YPES,')
CALL UPRINT (45.,10., '<F>IGURE 2 <C>USPRINT,')
CALL UEND
STOP
END
```

This is case shifting in GCS

Upper/Lower Case Shifting

Figure 3 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('ACENTER')
CALL USET ('LARGE')
CALL UPRINT (58.,30.,'THIS IS CASE SHIFTING IN <GCS,>')
CALL UPRINT (58.,28.,'UPPER/LOWER <C>ASE <S>HIFTING,')
CALL UPRINT (58.,15.,'FIGURE 3 <(USPRINT),>')
CALL UEND
STOP
END
```

# NOT ORIENTED ORIENTED 20 DEGREES

Character Orientation

Figure 4 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',',')
CALL USET ('SOFTWARE')
CALL USET ('ACENTER')
CALL UPRINT (50.,50.,'NOT ORIENTED,')
CALL UPSET ('ORIENTATION',20.)
CALL UPRINT (50.,40.,'ORIENTED 20 DEGREES,')
CALL USET ('HARDWARE')
CALL USET ('LARGE')
CALL UPRINT (50.,25.,'CHARACTER <ORIENTATION,')
CALL UPRINT (50.,20.,'FIGURE 4 <USPRINT,')
CALL UEND
STOP
END
```



*Italics*

No Italics

REFLECTIONS

Examples of Italization

Figure 5 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',';')
CALL USET ('ITALICS')
CALL USET ('SOFTWARE')
CALL UPRINT (25.,50.,'I>TALICS;')
CALL USET ('NOITALICS')
CALL UPRINT (25.,40.,'<I>O <I>TALICS<')
CALL USET ('NOCENTER')
CALL UPRINT (25.,30.,'REFLECTIONS;')
CALL UCOSYS (25.,30.,1.,-1.,0.)
CALL UPSET ('SLANT',40.)
CALL USET ('ITALICS')
CALL UPRINT (0.,0.,'REFLECTIONS;')
CALL USET ('WORLD COORDINATES')
CALL USET ('HARDWARE')
CALL USET ('ACENTER')
CALL USET ('LARGE')
CALL UPRINT (50.,15.,'E>XAMPLES OF <I>TALICIZATION;')
CALL UPRINT (50.,10.,'<F>IGURE 5 <USPRINT;')
CALL UEND
STOP
END
```

This is  
an exa  
mple of  
margin  
ing.

Margining  
Figure 6 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',',')
CALL USET ('PERCENT UNITS')
CALL UMARGN (45.,55.,30.,50.)
CALL USET ('LARGE')
CALL USET ('DEVICE')
CALL UPRINT (40.,45., 'THIS IS AN EXAMPLE OF MARGINING.,')
CALL USET ('ACENTER')
CALL UMARGN (0.,100.,0.,100.)
CALL UPRINT (50.,15., '<M>ARGINING,')
CALL UPRINT (50.,10., '<F>IGURE 6 <USPRINT,')
CALL UEND
STOP
END
```

# MORE MARGINING

Vertical Spacing by using Margining  
Figure 7 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('PERCENT UNITS')
CALL USET ('DEVICE')
CALL UMARGN (50.,50.5,20.,00.)
CALL USET ('LARGE')
CALL UPRINT (50.,00., 'MORE MARGINING,')
CALL USET ('ACENTER')
CALL UMARGN (0.,100.,0.,100.)
CALL UPRINT (50.,15., 'VERTICAL <S>PACING BY USING <M>ARGINING,')
CALL UPRINT (50.,10., '<F>IGURE 7 <USPRINT>,'')
CALL UEND
STOP
END
```

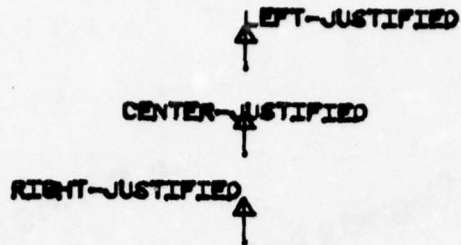
ndowing of hardware ch  
ng of software chare

# Examples of Windowing of Characters

Figure 8 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',',','')
CALL USET ('PERCENT UNITS')
CALL UDAREA (35.,65.,35.,65.)
CALL UOUTLN
CALL USET ('EXTRALARGE')
CALL UPRINT (-18.,68.,'INDOWING OF HARDWARE CHARACTERS,')
CALL USET ('SOFTWARE CHARACTERS')
CALL USET ('ACENTER')
CALL UPRINT (45.,49.,'INDOWING OF SOFTWARE CHARACTERS,')
CALL USET ('DEVICE')
CALL USET ('HARDWARE CHARACTERS')
CALL USET ('LARGE')
CALL UPRINT (59.,25.,'XAMPLES OF <W>INDOWING OF <C>HARACTERS')
CALL UPRINT (59.,28.,'IGURE 8 <USPRINT>,'')
CALL UEND
STOP
END
```





#### Justification Options

Figure 9 (USPRNT)

```
CALL USTART
CALL UPSET ('TERMINATOR','')
CALL USET ('LARGE')
CALL USET ('LARROW')
CALL UMOVE (58.,45.)
CALL UPEN (58.,58.)
CALL UPRINT (58.,58., 'LEFT-JUSTIFIED,')
CALL USET ('CJUSTIFICATION')
CALL UMOVE (58.,35.)
CALL UPEN (58.,48.)
CALL UPRINT (58.,48., 'CENTER-JUSTIFIED,')
CALL USET ('RJUSTIFICATION')
CALL UMOVE (58.,25.)
CALL UPEN (58.,38.)
CALL UPRINT (58.,38., 'RIGHT-JUSTIFIED,')
CALL USET ('CJUSTIFICATION')
CALL UPRINT (58.,15., 'JUSTIFICATION <O>PTIONS,')
CALL UPRINT (58.,18., '<F>IGURE 9 <USPRNT>,'')
CALL UEND
STOP
END
```

3D Character String

3D Character String 2

3D Character String 1

3D Character String 2

3D Software Character String Orientation

Figure 11 (USPRINT)

```
CALL USTART
CALL UPSET ('TERMINATOR',',',')
CALL USET ('LARGE')
CALL USET ('ACENTER')
CALL UPRINT (50.,15.,'3D SOFTWARE <C>HARACTER <S>TRING <O>RIEN
&TATION,')
CALL UPRINT (50.,10.,'<F>IGURE 11 <C>USPRINT,')
CALL USET ('SOFTWARE CHARACTERS')
CALL UUNDO (-100.,100.,-100.,100.)
CALL USPRINT (0.,0.,0.,'3D <C>HARACTER <S>TRING 1,')
CALL USPRINT (0.,-30.,-50.,'3D <C>HARACTER <S>TRING 2,')
CALL USCSYS (-30.,40.,30.,1.,1.,1.,45.,75.,45.)
CALL USPRINT (0.,0.,0.,'3D <C>HARACTER <S>TRING 3,')
CALL USCSYS (50.,50.,-10.,-1.,-1.,1.,0.,0.,-20.)
CALL USPRINT (0.,0.,0.,'3D <C>HARACTER <S>TRING 4,')
CALL UEND
STOP
END
```

**FUNCTION:**

This routine builds a new coordinate system with origin at the current pen position rotated by the specified amount about each axis.

**CALLING SEQUENCE:**

**CALL U3ROTA(RX,RY,RZ)**

Where

**RX,RY,RZ** indicates the amount of rotation around each of the axes in current angular units.

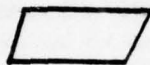
**OPTIONS:**

Rotation Application Order: See U3CSYS

**COMMENTS:**

This routine causes a new coordinate system to be built with origin at the current beam position and with a scale factor of 1 along each axis. A full description of the user coordinate system facility can be found in the U3CSYS writeup and in the UCOSYS writeup.

**Programming Notes:**



```
CALL USTART
CALL UMOVE (50.,75.)
CALL UROTAT (0.)
CALL UPLYGN (0.,0.,3.,10.)
CALL UROTAT (180.)
CALL UPLYGN (0.,0.,3.,10.)
CALL USET ('WORLD COORDINATES')
CALL UMOVE (25.,25.)
CALL USROTA (75.,0.,0.)
CALL UPLYGN (0.,0.,4.,10.)
CALL USET ('WORLD COORDINATES')
CALL UMOVE (75.,25.)
CALL USROTA (0.,45.,0.)
CALL UPLYGN (0.,0.,4.,10.)
CALL UEND
STOP
END
```



**FUNCTION:**

This routine composes a new coordinate system with the origin at the current beam position and scaled as specified along each axis.

**CALLING SEQUENCE:**

**CALL U3SCAL(SX,SY,SZ)**

Where

**SX,SY,SZ** are the multiplicative scale factors along each axis respectively.

**OPTIONS:**

None.

**COMMENTS:**

The rotation factors used in composing the new scaled coordinate system are 0. around each axis. See the U3CSYS and UCOSYS writeup for a full description of the user coordinate system facility.

**Programming Notes:**

**FUNCTION:**

This routine returns to the user the limits of his 3-D U3WINDO and U3AREA.

**CALLING SEQUENCE:**

**CALL U3STUD (ARRAY)**

Where

**ARRAY** is an array of at least twelve words to contain the current settings.

**OPTIONS which may apply:**

Device Units: 'INCHES', 'CENTIMETERS', 'PERCENT', 'FONT', 'RASTER', 'SPECIFICATION'

**COMMENTS:**

A call to this routine will return to the user the limits of his 3-D virtual window and display area in the order:

ARRAY (1) = virtual X minimum boundary  
ARRAY (2) = virtual X maximum boundary  
ARRAY (3) = virtual Y minimum boundary  
ARRAY (4) = virtual Y maximum boundary  
ARRAY (5) = virtual Z minimum boundary  
ARRAY (6) = virtual Z maximum boundary  
ARRAY (7) = device X minimum boundary  
ARRAY (8) = device X maximum boundary  
ARRAY (9) = device Y minimum boundary  
ARRAY (10) = device Y maximum boundary  
ARRAY (11) = device Z minimum boundary  
ARRAY (12) = device Z maximum boundary

To recover only the 2-D boundaries, see USTUD.

**Programming Notes:**

**FUNCTION:**

This routine returns the three-dimensional coordinates of the pen position in current units.

**CALLING SEQUENCE:**

**CALL U3WHER(X,Y,Z)**

Where

**X,Y,Z** will contain the pen position coordinates in current units upon exit from this rotation.

**OPTIONS:**

Pen Coordinate Options: see U3PEN

**COMMENTS:**

The coordinates returned will always indicate 'ABSOLUTE' position in the current coordinate system since the 'RELATIVE' position is always (0,0,0).

**Programming Notes:**

**FUNCTION:**

This routine specifies the boundaries of the user window in virtual 3-space.

**CALLING SEQUENCE:**

**CALL U3WENDO(XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)**

Where

<b>XMIN</b>	is the minimum X-boundary of the desired new window.
<b>XMAX</b>	is the maximum X-boundary of the desired new window.
<b>YMIN</b>	is the minimum Y-boundary of the desired new window.
<b>YMAX</b>	is the maximum Y-boundary of the desired new window.
<b>ZMIN</b>	is the minimum Z-boundary of the desired new window.
<b>ZMAX</b>	is the maximum Z-boundary of the desired new window.

**OPTIONS which may apply:**

Z-axis clipping: 'NOZCLIPPING', 'ZCLIPPING'

**COMMENTS:**

The concept of windowing into virtual space expands greatly the capabilities of the user. In three-dimensional space, the user is assumed to be located somewhere in virtual space (i.e., at the view point) and looking in some direction (i.e., toward the view site) (see UVIEW). Normally, he is only interested in a particular subset of all the space within his field of view. This subset is specified by the U3WENDO routine. The portion of the user's picture which is within the boundaries of the window will be projected onto a plane perpendicular to the viewing vector and in front of the viewer. This plane is known as the view plane (See UVWPLN). The X and Y window boundaries refer to lines on the view plane which form a rectangle. The Z window boundaries indicate the portion of space in front of the viewer to be projected onto the plane if 'ZCLIPPING' has been specified. Normally, 'NOZCLIPPING' is specified, in which case all lines which are projected onto the view plane within the X and Y window boundaries are visible.

Subroutine U3WENDO will generate an error condition if the maximum boundary is specified less than or equal to the minimum boundary and the previous setting of the window will be retained.

**Programming Notes:**



**FUNCTION:**

This routine displays textual output at the pen position specimen in three-space. The pen will be repositioned to the location it was at prior to the call to U3WRIT.

**CALLING SEQUENCE:**

**CALL U3WRIT(X,Y,Z,DATA)**

Where

**X,Y,Z** are the coordinates of the beginning of the alphanumeric output in current units.

**DATA** is a GCS Hollerith character string terminated by the GCS terminator character if 'TEXT' is specified.

is a REAL variable or literal if 'INTEGER' or 'REAL' is specified.

is a REAL array of two elements if 'XYCOORDINATES' is specified.

is a REAL array of three elements if 'XYZCOORDINATES' is specified.

**OPTIONS:**

Coordinate Space Options:  
Alphanumeric Character type:  
Alphanumeric Output type:  
Alphanumeric Spacing Option:

See U3PRNT  
See U3PRNT  
See U3PRNT  
See U3PRNT

**COMMENTS:**

This routine only differs from U3PRNT in the position of the pen at return from the routine. U3PRNT leaves the beam positioned at the end of the text output. U3WRIT returns the pen to the position it held before entry to U3WRIT. For a full discussion of the various U3WRIT options, see U3PRNT.

**Programming Notes:**